MaxonC++

Das Entwicklungssystem zur objektorientierten Programmierung

AMIGA

MakeAPP

Für anwenderfreundliche Programmgestaltung

MAXON computer

Inhaltsverzeichnis

Vorwort 7

Einleitung 8

Allgemeines 9

Was ist eine Benutzeroberfläche? 10

Start und Aufbau des Programms 11

Die Bedienung durch Popup-Menüs 11

Die Programm-Requester 12

Das Dateiauswahlfenster 12

Der Farbrequester 13

Der Text-Requester 14

Der Namens-Requester 14

Der Alert-Requester 15

Das Konstruieren 15

Die Menüs in der Objektauswahl 15

Neuanlegen: 15
Laden: 16
Hinzuladen: 16
Speichern : 16
Speichern als: 16
Objekt speichern: 16
Beenden : 16
Menüleiste hinzufügen: 17
Name ändern : 17
Objekt löschen: 17
MakeAPP Setup: 17
Auflösung: 18
Ausgabe: 18

Requester 19

Das Requester-Popup-Menü 19
- Requester-Objekte hinzufügen 19
- Requester Farbe wählen 19
- Gadgets sortieren 19
- Requester Auflösung ändern 20
- Requester testen 20
- Objekte im Clipboard 20
- Zurück zur Objektauswahl 20

Requester vergrößern 21

Requester Texte bearbeiten 21

Positionieren von Requestertexten 21

Texte verändern 21
- Texte löschen 22
- Texte kopieren 22
- Texte ins Clipboard kopieren 22
- Texte benennen 22
- Texte zentrieren 22

Gadgets bearbeiten 23

Boolean Gadgets: 23

String Gadgets 23

Proportional Gadgets 23

Gadgettexte positionieren und verändern 24

Gadget-Flags einstellen 24
- Flags für Boolean Gadgets 24
- Gadgets kopieren und löschen 26
- Gadgets benennen 26
- Gadgets ins Clipboard kopieren 26
- Gadgetgrößen anpassen 27
- Füllmusterauswahl für Boolean Gadgets 27
- Gadget- und Gadgettextzentrierung 27
- Gadget Border verändern 27
- Neue Border hinzufügen 28
- Border Clipboard 28
- Border und Borderpunkte löschen 28

Gestaltungshilfmittel 28
- Symmetrisch editieren 29

Tastatursteuerung 29

Images in Gadgets 29

Laden, Auswählen und Löschen von Grafiken 30

Ausschnitt festlegen 30

Farben übernehmen und Image konvertieren 30

Requester-Border-Editor 30

Requester-Border positionieren 31
Border löschen und kopieren 31
Border zentrieren und Farbe einstellen 31
Border editieren 31

Menüleisten 32
Menutiteltexte ändern 32
Anbringen des Pull-Down-Menüs 33

Verschieben von Titeln, Einträgen und Untereinträgen 33

Vergrößern von Einträgen und Untereinträgen. 33
Texte verschieben und ändern 34

Menü-Flags 34
Menü testen 35
Menüs justieren 36

Tastaturbelegungstabelle 37

Vorwort

Im Zuge der Neuerungen, die Commodore seinem Betriebssystem für den AMIGA zugute hat kommen lassen, werden immer mehr Programme umgeschrieben und auf das neue Aussehen und die neu enthaltenen Funktionen angepaßt. Leider funktionieren dann solche Programme nur noch mit dem neuen Betriebssystem, weshalb viele Software-Entwickler Teile des Systems Nachprogrammieren, um zu gewährleisten, daß ihre Software auch noch unter älteren System-Versionen gestartet werden kann.
Meist betrifft dies das Aussehen, denn unter System 2.X sieht alles plastisch aus und wirkt nicht so unprofessionell wie in älteren Versionen.

Das Aussehen ist sicher auch das, was Ihnen beim MakeAPP zuerst auffallen muß. Es präsentiert sich im schicken "2.0 Look". Aber anders als bei anderen Programmen werden die ganze Benutzeroberfläche, Requester und Menüleisten nach dem Programmstart nachgeladen. Damit ist nun auch im MakeAPP die bereits mit der ersten Version angepriesene Philosophie der leichten Änderbarkeit der Oberfläche verwirklicht. Die Oberfläche ist komplett mit dem MakeAPP generiert und in einer Ressource-Datei zusammengefaßt. Sollte ich nun irgendwelche Änderungen vornehmen müssen, vielleich noch ein zusätzliches Gadget in den einen oder anderen Requester oder einen weiteren Menüpunkt, lade ich einfach meine Oberfläche und verändere sie mit dem MakeAPP. Ich muß nicht mit komplexen Strukturen hantieren, um mir nach einem Compilerlauf das Ergebnis meiner Veränderungen ansehen zu können.

Für diejenigen, die dennoch lieber mit endlosen Datenwüsten programmieren möchten, wurde der Source-Code-Generator überarbeitet und erweitert. Es kann nun Source-Code für Basic (AMIGA, GFA, HiSoft), C, Assembler, KickPascal und sogar Modula 2 erzeugt werden. Für jeden ist etwas dabei.

Der File-Requester hat sich geändert und ist nun in der MakeAPP-Library enthalten. Er ist komplett reentrant programmiert und dadurch von mehreren Programmen gleichzeitig nutzbar. Durch das Library-Konzept ist es auch vollkommen gleichgültig in welcher Programmiersprache Sie programmieren.
Da all seine Funktionen auch über die Tastatur erreichbar sind, ist auch für Textverarbeitungen und Texteditore bestens geeignet.

Viel Spaß beim Programmieren

Ihr Jürgen Haage

Einleitung

Ein selbstgeschriebenes Programm sollte nicht nur funktionstüchtig und fehlerfrei, sondern auch leicht bedienbar sein.
Das beste Hilfsmittel für die leichte Bedienbarkeit ist die Benutzeroberfläche, die der Amiga zur Verfügung stellt. Sie heißt Intuition und enthält alle Funktionen, die durch Anwendungen mit der Maus erreicht werden können. Die wohl bekanntesten Aufgaben sind Screens und Fenster, die sehr vielseitig mit der Maus manipuliert werden können. Sie stellen als Ein- und Ausgabeeinheiten sehr flexible Bildschirmobjekte dar.

Ein weiteres Hilfsmittel sind Requester. Dies sind Bildschirmformulare, in denen meist Eingaben oder Entscheidungen verlangt werden. Das beste Beispiel hierfür ist der Preferences Requester, in dem man die Systemeinstellungen des Amiga vornehmen kann.

Auf einem Requester stehen sogenannte Gadgets, die bei Selektierung mit der Maus bestimmte Programmreaktionen verursachen. Durch Gadgets kann der Programmierer sehr schöne Effekte in seinem Programm erzielen. Sie können beispielsweise Bilder enthalten, die sich bei Selektierung verändern, so daß der Anwender bildlich über die eingestellten Programmfunktionen informiert wird.

Die Benutzung von Gadgets ähnelt der von Menüleisten. Durch die Selektierung der Gadgets kann, ebenso wie bei der Auswahl von Menüeinträgen, zu bestimmten Programmfunktionen verzweigt werden.

Dennoch wird in manchen Fällen der Einsatz von Requestern bevorzugt. Das ist zum Beispiel dann der Fall, wenn Eingaben und Einstellungen gleichzeitig vom Anwender verlangt werden. Um Texteingaben in Requestern machen zu können, werden sehr komfortable Eingabegadgets eingesetzt. Ihre Größe, das heißt der auf dem Bildschirm angezeigte Text, ist zwar auf die Requestergröße begrenzt nicht aber die Gesamttextlänge, die eingegeben werden kann. Hat man durch die Eingabe die Gadgetbegrenzung erreicht, wird der Text im Gadget gescrollt. Mit den Cursortasten kann der Text nach links oder rechts gescrollt und Einfügungen vorgenommen werden.

In Menüleisten sind Eingabefelder nicht möglich. Ihre Hauptaufgabe ist es, im Programm zu einzelnen Funktionen zu verzweigen. Es besteht allerdings auch die Möglichkeit, Einstellungen über einen Menüeintrag zu mrealisieren. Zur Kennzeichnung wird ein Häkchen vor den Menüeintrag gestellt. Beim Anwählen des Menüs sieht der Anwender sofort die vorgenommenen Einstellungen.

Die Entscheidung, ob Menüleisten oder Requester zur Anwendungsvereinfachung eingesetzt werden, bleibt dem Programmierer überlassen. Mit herkömmlichen Program-

miersprachen wird sich der Programmierer genau überlegen, ob er für einen Requester nicht doch auch eine Menüleiste einsetzen kann, da hierbei, bedingt durch die komplexen grafischen Gestaltungsmöglichkeiten, mit sehr komplizierten Strukturen umgegangen werden muß. Dies bedeutet, mehr Zeit in ein Programm investieren zu müssen, da sehr vieles einfach ausprobiert und wieder geändert werden muß.

Das Experimentieren enfällt, wenn man seine Requester und Menüleisten direkt am Bildschirm gestalten kann. Dabei hilft MakeAPP. Programmtechnisch sehr aufwendig gestaltete Requester lassen sich in sehr kurzer Zeit erstellen und in eigene Programme übernehmen. Das Programm enthält selbst Requester und Menüleisten, um dem Anwender die Arbeit mit dem Programm zu vereinfachen und auch Anregungen für eigene Gestaltungen zu vermitteln.

Allgemeines

Das Ihnen vorliegende Handbuch ist so konzipiert, daß nach dem Durcharbeiten jeder, egal ob Anfänger, Fortgeschrittener oder Profi, in der Lage ist, für seine Programme die Benutzeroberfläche schnell und komfortabel zu generieren und zu integrieren.

Das MakeAPP ist vollkommen mausgesteuert und weitgehend selbsterklärend. Die Voraussetzung zur richtigen Anwendung des Programms ist die Kenntnis des Amiga-Benutzerhandbuchs und von Intuition-Objekten selbst.

Requester und Menüleisten können zwar auf dem Bildschirm gestaltet werden, die Verwaltung übernimmt aber das Programm, für das sie vorgesehen sind. Das heißt, die Art wie beispielsweise auf eine Gadgetselektierung reagiert werden soll, ist abhängig von den Flags, die zu den jeweiligen Gadgets eingestellt wurden. Die Flags haben fest vorgegebene, englische Bezeichnungen. Die Einstellung im MakeAPP wird über Requester vorgenommen, in denen die Bezeichnungen der Flags einfach mit der Maus ausgewählt werden können. Kennt der Programmierer die Reaktionen von Intuition auf die jeweiligen Flags, weiß er auch, wie sich die einzelnen Gadgets im Programm verhalten.

Um die Möglichkeiten des MakeAPP voll ausschöpfen zu können, sollten Sie nicht sofort beginnen, mit dem Programm zu arbeiten.

Die Bedienung eines Programms erlernen Sie am schnellsten und effektivsten, indem Sie sich das Handbuch Kapitel für Kapitel durchlesen und alle beschriebenen Funktionen nachvollziehen, bis alle Zusammenhänge klar werden.

Was ist eine Benutzeroberfläche?

Prinzipiell kann man jegliche Art von Menüführungen in einem Programm als Benutzeroberfläche bezeichnen. Eine Menüauswahl per Funktionstasten, zu der man durch eine Textanzeige aufgefordert wird, ist ebenso eine Benutzeroberfläche wie ein Befehlsinterpreter.

Programme, die durch reine Textdialoge mit dem Anwender kommunizieren gelten spätestens seit erscheinen der ATARI ST und der Commodore AMIGA Computer als veraltet. Es sind die ersten Computer im Heimbreich, denen eine grafische Benutzeroberfläche mitgeliefert wird. Beim Amiga besticht sie durch die Farbenvielfalt und Verspieltheit. Beim Atari durch die Ähnlichkeit zur Benutzeroberfläche des Apple Macintosh. Die fehlende grafische Benutzerführung bei älteren Computer liegt zum einen beim erheblichen Speicheranspruch und zum anderen an der geringen Rechenleistung der Hardware.

Start und Aufbau des Programms

Die Benutzeroberfläche von MakeAPP ist mit MakeAPP selbst entwickelt. Zur Verwaltung der Oberfläche dient die RCT.LIBRARY.

Die Installation von MakeAPP und der Rct.Library geschieht ebenfalls bei der Installation des MaxonC++-Entwicklungssystems vollkommen automatisch.

Die Library enthält die Funktionen zum Laden der Oberfläche und zur Verwaltung aller auf der Oberfläche zur Anzeige gebrachten Objekte. Aber auch so wichtige Bestandteile wie das Dateiauswahl- und das Alarmfenster.

Nach dem Start erscheint auf dem Monitor das Objektauswahlfenster von MakeAPP. Bevor Sie jetzt zu experimentieren beginnen, sollten Sie die nachfolgenden Kapitel des Handbuchs genau durchlesen. Gerade die Bedienung einzelner Programmelemente zu verstehen, ist sehr wichtig und nicht durch Versuche zu erreichen.
Das Programm ist in drei Teile untergliedert. Im ersten Teil befinden Sie sich nach dem Starten des Programms in der Objektauswahl. Hier können Funktionen wie Speichern, Laden, Einstellungen und Requester- oder Menüauswahl ausgeführt werden. Von der Objektauswahl können Sie in den Requester- oder Menü-Editor verzweigen. Der Requester-Editor ist der zweite Programmteil. Wie der Name schon sagt, können hier Requester editiert werden. Im dritten Programmteil, dem Menü-Editor kann man Amiga-Menüleisten konstruieren.

Die Bedienung durch Popup-Menüs

Die Bedienung des Requester- und Menü-Editors unterscheidet sich von der der Objektauswahl, durch die Art der eingesetzten Auswahlmenüs. In der Objektauswahl werden herkömmliche Amiga-Menüs eingesetzt. Die Kenntnis darüber, wie solche Menüs bedient werden, wird vorausgesetzt.

In den Editor-Programmteilen werden sogenannte Popup-Menüs eingesetzt. Der Unterschied zu normalen Menüleisten liegt darin, daß Popup-Menüs direkt angezeigt werden, wenn der rechteMausknopf gedrückt wird. Im allgemeinen werden Popup-Menüs direkt auf der aktuellen Mausposition geöffnet, was lästiges Hin- und Herfahren mit der Maus erübrigt. Zu Anfang ist diese Art des Umgangs mit dem Menü sicher etwas gewöhnungsbedürftig. Aber hat man sich einmal daran gewöhnt, möchte man die Popups in keinem Fall mehr missen.

Die Auswahl der Menüeinträge geschieht genau wie bei herkömmlichen Menüs, indem Sie den entsprechenden Eintrag mit der Maus anwählen und die Menütaste der Maus loslassen.

Während der Darstellung eines Popup-Menüs werden die Farben des Screens verändert, um alle Menüs in einheitlichen Farben zu halten. Dies ist besonders wichtig, da bei der Requestergestaltung Grafiken eingesetzt werden können, die meist eine eigene Farbpalette verwenden. Bei Grafiken mit einer fein abgestuften Farbpalette würde im ungünstigsten Fall kein Menüeintrag lesbar sein.

Nach Loslassen des Menüknopfes wird die ursprüngliche Farbpalette wieder eingestellt. Menüeinträge, deren Texte in blauer Farbe angezeigt werden, können bei der jeweils durchgeführten Aktion nicht ausgewählt werden. Diese werden erst gar nicht selektiert, wenn man den Mauspfeil über sie bewegt. Einige Popup-Menüs enthalten Extra-Funktionen, die in dem jeweiligen Kapitel näher erläutert werden.

Die Programm-Requester

Das Konstruieren von Menüleisten oder Requestern wird durch Requester erleichtert, aus denen Sie das Gewünschte mit der Maus auswählen können.
Dabei gibt es Requester, die sowohl im Requester-Editor als auch im Menü-Editor oder in der Objektauswahl angezeigt werden.
Im nachfolgenden Abschnitt wird die Funktion der globalen Requester näher erläutert und auf die jeweiligen Programmteile verwiesen, aus denen sie aufgerufen werden können.

Das Dateiauswahlfenster

Vor den meisten Zugriffen auf Diskettenlaufwerke oder Harddisks wird der File-Requester angezeigt. In diesem Requester können Sie Verzeichnisse und Dateien mit der Maus auswählen. Im MakeAPP wird er benutzt, um Ressource-Dateien zu speichern oder zu laden.

Der File-Requester ist in der Rct-Library enthalten und vollkommen reentrant, so daß er auch gleichzeitig von anderen Programmen benutzt werden kann.

Selbst wenn Sie in BASIC programmieren ist es kein Problem den File-Requester für ihre eigenen Programe zu verwenden.

Da er komplett mit der Tastatur gesteuert werden kann, ist er auch bestens für einen Texteditor oder eine Textverarbeitung geeignet. Bei vielen Editoren muß zur Auswahl einer Datei, zum Speichern oder Laden, immernoch zur Maus gegriffen werden. Dies entfällt mit den Rct-File-Requester.

Die genau Erklärung zum File-Requester finden Sie in der Referenz zur RCT-Funktionsbibliothek.

Der Farbrequester

Im Farbrequester können Sie die Farben für Requester,Texte, Gadgets und Border einstellen. Er ist sowohl aus dem Requester-Editor als auch aus dem Menü-Editor erreichbar.

Je nach eingestellter Auflösung werden die entsprechenden Farben angezeigt und können mit Klicken ausgewählt werden. Über die Schieberegler läßt sich der Farbwert der ausgewählten Farbe verändern, wenn man Requester konstruiert. Im Menü-Editor ist diese Möglichkeit nicht gegeben, da zu einem Menü keine entsprechende Farbpalette angezeigt werden kann.

Für Requester- und Menüleistentexte ist es wichtig, den Zeichenmodus angeben zu können. Dazu stehen im Requester die Zeichenmodi-, Vorder- und Hintergrund-Gadgets.

JAM1 Hierbei wird nur die Vordergrundfarbe als Zeichenfarbe gewünscht.
JAM2 Die Vorder- und Hinergrundfarbe kann eingestellt werden.
XOR Die Bildpunkte des Textes werden exclusiv oder mit der Hintergrundfarbe verknüpft.
INVERS Der Text wird in der Vordergrundfarbe invers dargestellt.

Vordergrund
Die Einstellung für die Vordergrundfarbe.

Hintergrund
Die Einstellung für die Hintergrundfarbe.

Im Programm übernehmen ?

Dieses Gadget ist nur im Requester-Editor aktiv. Selektiert man es, wird die eingestellte Farbpalette für den Requester im eigenen Programm auch angezeigt. Anderenfalls wird der Requester unabhängig von der im MakeAPP eingestellten Farbe angezeigt.
„OK" übernimmt die eingestellte Farbe.
„Abbruch" macht alle Eingaben wieder rückgängig und behält die alte Einstellung bei.

Der Text-Requester

In MakeAPP ist es möglich, Requester-, Gadget- und Menüeintragtexten unterschiedliche Zeichensätze und Farben zuzuordnen. Dazu wird beim Editieren von Texten der Text-Requester angezeigt. Mit den Auswahl-Gadgets in diesem Requester können Sie Zeichensätze laden, und die Textfarbe und Zeichenmodi auswählen.

Zur Zeichensatzauswahl bedient sich MakeAPP, des in der Ressource-Bibliothek enthaltenen Font-Requesters.

Der Namens-Requester

Über die Namen, die Sie den MakeAPP-Objekten geben, können Sie in Ihrem Programm nachträglich die Objekte beeinflussen. Bei Gadgets und Menüleisten ist dies besonders wichtig, da Sie durch die angegebenen Namen die Nachrichten, die von der Funktion R_GetMsg() zurückgeliefert werden, besser entschlüsseln können.

Wenn Sie neue Requester oder Menüleisten konstruiert, müssen Sie unbedingt einen Name angegeben. Nur über diesen Namen können Sie im Programm das Menü oder den Requester anzeigen. Der Name darf keine Leerzeichen oder Umlaute enthalten.
Das MakeAPP speichert diese Variablen in der Datei mit der Endung „.H".

Der Alert-Requester

Dieser Requester wird immer bei Sicherheitsabfragen angezeigt. Dies ist zum Beispiel der Fall, wenn Requester oder Menüleisten aus dem Speicher gelöscht werden sollen (siehe Ressource-Bibliothek R_FormAlert).

Das Konstruieren

In der Objektauswahl, in der Sie sich direkt nach dem Start von MakeAPP befindet, können Sie neue Requester oder Menüleisten konstruieren oder bereits konstruierte einladen und verändern. Für jede neu konstruierte Menüleiste oder Requester wird im Fenster ein Image angezeigt. Unter den Images steht der Name, den Sie im Namens-Requester für das jeweilige Objekt angegeben haben.

Das Image des zuletzt veränderten Objektes wird invertiert angezeigt und ist gleichzeitig das aktuelle für alle weiteren Funktionen, die in der Objektauswahl stattfinden. Wird das aktuelle Objekt mit der Maus angeklickt, öffnet sich, je nach dem, ob es sich um ein Menü- oder Requester-Image handelt, der jeweilige Editor, in dem das bereits konstruierte Objekt angezeigt wird. Bei Klicken auf ein anderes Image, wird das angeklickte zum aktuellen gemacht und invertiert dargestellt. Bei einem weiteren Klicken auf das gleiche Image wird der entsprechende Editorscreen geöffnet und das Objekt dargestellt.

Die Menüs in der Objektauswahl

Aus dem Menü „Projekt" können Sie Funktionen aufrufen, um MakeAPP-Dateien zu laden oder zu speichern und um das Programm zu beenden.

Neuanlegen:

löscht alle konstruierten Menüleisten und Requester aus dem Speicher und die Images im Objektauswahlfenster. Die zuvor erscheinende Sicherheitsabfrage verhindert, daß zufällig gelöscht wird. Es passiert überhaupt nichts, wenn keine Objekte zum Löschen vorhanden sind.

Laden:

Aus dem angezeigten Standart-File-Requester kann die Datei ausgewählt werden, die geladen werden soll. Sind bereits Menüleisten oder Requester konstruiert worden, werden diese vor dem Laden aus dem Speicher gelöscht. Nach dem Laden werden die Images der gefundenen Objekte im Objektauswahlfenster angezeigt.

Hinzuladen:

Wie bei „Laden" erscheint auch beim Hinzuladen der Standart-File-Requester. Hierbei werden bereits konstruierte Objekte im Speicher gehalten und die neugeladenen Objekte dazugestellt. Sind für die Objekte, in den zu ladenen Dateien, Namen definiert worden, werden diese gelöscht um doppelte Namen zu vermeiden. Für jeden doppelten Requester- oder Menüleistennamen wird ein neuer Name verlangt.

Speichern :

die konstruierten Objekte werden unter dem aktuellen Dateinamen auf der Diskette gespeichert. Dabei wird zur Sicherheit noch einmal nachgefragt, ob die Funktion beabsichtigt wurde. Existiert kein aktueller Dateiname, wird der Datei-Auswahl-Requester angezeigt.

Speichern als:

Im Gegensatz zu „Speichern" wird bei Auswahl dieser Funktion immer der Datei-Auswahl-Requester angezeigt und Sie können einen neuen Dateinamen angeben. Existiert die angegebene Datei bereits, werden Sie zusätzlich noch gefragt, ob die Datei überschrieben werden soll.

Objekt speichern:

Mit dieser Funktion wird das aktuelle (invertierte Image) Objekt gespeichert. So können auch ganze Menüleisten oder Requester kopiert werden. Man speichert das Objekt mit der Funktion „Objekt speichern" auf Diskette und lädt es anschließend über den Menüpunkt „Hinzuladen" wieder ein.

Beenden :

Über diesen Menüpunkt können Sie die Arbeit mit dem MakeAPP beenden. Dabei findet immer eine Sicherheitsabfrage statt, um zu vermeiden, daß die Funktion unbeabsichtigt angewählt werden kann, ohne sie rückgängig machen zu können.

Über den zweiten Menüpunkt „Objektbearbeitung" sind die Funktionen erreichbar, mit denen neue Requester und Menüleisten angelegt, gelöscht und Einstellungen vorgenommen werden können. Requester hinzufügen: Durch Auswahl dieses Menüpunktes können Sie neu zu konstruierende Requester anmelden. Dazu erscheint der Namens-Requester, in den der Name des Requesters eingegeben werden muß. Tritt dabei ein Fehler auf (doppelter Name, Leerzeichen), flackert der Bildschirm kurz und die entsprechende Fehlermeldung erscheint in der Screentitelzeile. Wird der Name angenommen, öffnet sich der Requester-Editorscreen, und Sie können mit dem Konstruieren beginnen.

Menüleiste hinzufügen:

über diesen Menüpunkt können Menüleisten neu angelegt werden. Wie im Menüpunkt „Requester hinzufügen" wird auch hier der Namens-Requester angezeigt. Ist der angegebene Name korrekt, wird der Menü-Editorscreen geöffnet.

Name ändern :

Der im Objektauswahlfenster unter den Images stehende Name kann mit dieser Funktion geändert werden. Im angezeigten Namens-Requester steht dann der bereits angegeben Name.

Objekt löschen:

Bevor das aktuelle Objekt (invertiertes Image) aus dem Speicher gelöscht wird, werden Sie im Alert-Requester gefragt, ob das Objekt tatsächlich gelöscht werden soll.

MakeAPP Setup:

In dem angezeigten Setup-Requester können Sie Einstellungen für die Handhabung von Popup-Menüs und Clipboards machen.

Die obere Box des Requesters enthält die Einstellmöglichkeiten für Clipboards. Dies sind Merker, in die Requester- und Menüleisten- Objekte (Gadgets, Menüeintrage...) gestellt werden können, um sie zu einem späteren Zeitpunkt eventuell wieder holen zu können.
Mit den ersten beiden Tastern wird eingestellt, ob eine Sicherheits- abfrage gemacht werden soll, wenn Gadgets, Texte, Border oder Menüeinträge gelöscht werden. Alle Löschvorgänge, die auf die genannten Objekte ausgeführt werden, können auch wieder rückgängig gemacht werden. Ob das gewüscht ist, wird mit dem nächsten beiden Tastern eingestellen. „Ja" bedeutet, daß nach dem Löschen eines Objektes dieses im Clipboard zwischengespeichert wird. Wird das nächte Objekt gelöscht, so

überschreibt dies den alten Inhalt des Clipoards. Stellen Sie diese Abfrage auf „Nein", bleibt das Clipboard bei einem Löschvorgang unbeeinflußt.

Die Abfragen in der unteren Box des Requesters beziehen sich auf die Ausgabe der Popup-Menüs. Voreingestellt ist hierbei, daß diese in der linken oberen Screenecke angezeigt werden. Es gibt drei weitere Varianten, die zulassen, daß sie links vom Mauszeiger, direkt an der Position des Mauszeigers oder rechts vom Mauszeiger angezeigt werden. Dies muß jeder Anwender individuell entscheiden. Die Einstellungen können auf Diskette gespeichert werden.
Möchte man die Einstellungen nur ausprobieren, genügt es, den Requester mit Klicken auf das Gadget „Benutzen" zu verlassen. Bei „Abbruch" der Eingabe werden die alten Einstellungen wieder übernommen.

Auflösung:

Mit dieser Funktion wird die Auflösung für neu zu konstruierende Requester oder Menüleisten eingestellt. Dazu erscheint der Auflösungs-Requester.

Ausgabe:

Mit der Auswahl dieses Menüpunktes erscheint ein Requester in dem Sie die Ausgabeformate festlegen können. Es besteht die Möglichkeit, zusätzlich zur normalen Ressource-Datei eine komprimierte Datei schreiben zu lassen. Die normale Datei wird beim Speichern allerdings immer erzeugt, da in MakeAPP keine komprimierten Dateien geladen werden können. Desweiteren können Sie hier wählen, ob eine C++-Sourcecode-Datei und eine Datei mit Lokalen-Strings gespeichert werden soll.

Der komprimierten Ressource-Datei wird die Extension „.STRIP" angehängt. Die C++-Sourcecode-Datei erhält die Extension „.CPP" und der Datei mit den Lokalen-Strings wird „.RLOCALE" angehängt.

Requester

In den Requester-Editor gelangen Sie entweder durch Doppelklick auf ein Requester-Image im Objektauswahlfenster oder durch Hinzufügen eines neuen Requesters über die Objektauwahl-Menüleiste. In diesem Editor stehen Ihnen alle Hilfsmittel zur Verfügung, die Sie zur Gestaltung eines Requesters benötigen. Aus dem Requester-Popup-Menü können Gadgets, Texte und Border in den Requester gestellt werden, die Sie dann mit der Maus positionieren und bearbeiten können. Für jede Objektart erscheint ein eigenes Popup-Menü, in dem die Funktionen erreichbar sind, die Sie zur Bearbeitung benötigen. Das jeweils aktuelle Objekt ist invertiert auf dem Requester angezeigt. In der Titelleiste des Screens steht die Objektart, der Name des Objekts und die Position des Objekts innerhalb des Requester. Sind zum Beispiel mehrere Gadgets, Texte und Border im Requester enthalten, und Sie klicken mit der Maus ein Objekt an, wird entsprechend der Objektart der Screentitel und das Popup-Menü umgestellt und das Objekt invertiert angezeigt.

Das Requester-Popup-Menü

Dieses Menü ist immer erreichbar, wenn Sie mit der Maus auf den Requester oder außerhalb des Requesters klicken. Das vorher invers angezeigte aktuelle Objekt wird dann normal dargestellt.

Requester-Objekte hinzufügen

Über die Menüpunkte Text, Boolean Gadget, String Gadget, Prop Gadget und Border wird das jeweils ausgewählte Objekt zum Requester hinzugefügt und angezeigt. Neue Objekte stehen immer in der linken oberen Ecke des Requesters.

Requester Farbe wählen

Der Menüpunkt Farbe zeigt den Farb-Requester an, in dem Sie die gewünschte Requesterfarbe einstellen können. Die Gadgets für die Zeichenmodi sind bei der Requesterfarbwahl ausgeschaltet.

Gadgets sortieren

Bei großen Requestern die viele Gadgets beinhalten, dauert der Aufbau meist länger als eine Sekunde. Dabei ist die Aufbaureihenfolge der Gadgets im Requester sehr schön sichtbar. Diese Reihenfolge können Sie beeinflussen, indem der Sortiermodus

für Gadgets eingeschaltet wird. Vor dem Menüpunkt Sortieren wird dann ein Haken angezeigt. Klicken Sie nun in der gewünschten Reihenfolge die Gadgets in Ihrem konstruierten Requester an und schalten den Sortiermodus wieder aus. Die Gadgets werden in der Reihenfolge angezeigt, wie Sie sie mit der Maus angeklickt haben.

Requester Auflösung ändern

Wird der Menüpunkt Auflösung angewählt, erscheint der Auflösung-Requester, in dem Sie die Auflösung des Requester-Editorscreens ändern können. Die eingestellte Auflösung ist unabhängig von der Ihres späteren Programms. Sie dient lediglich zum besseren Konstruieren.

Requester testen

Der Requestertest ist zum Ausprobieren der Gadgetselektierungen, wie sie im Programm erscheinen. Jedes Gadget kann bei Anwahl in einer anderen Form selektiert werden. Die Erläuterung zur Einstellung der Selektierung finden Sie im Kapitel „Gadgets bearbeiten". Haben Sie den Menüpunkt „Test Requester" angewählt und klicken auf ein Gadget, erscheint die entsprechende Selektierung. Um weiter konstruieren zu können, muß die rechte Maustaste gedrückt werden.

Objekte im Clipboard

Vor den nächsten fünf Menüpunkten ist ein Image angezeigt. Dieses Image steht als Symbol für das Clipboard. Das Clipboard ist ein Zwischenbuffer für das jeweilige Objekt. Möchten Sie beispielsweise ein und das gleiche Gadget in mehreren Requestern anzeigen, müssen Sie dieses nicht immer neu konstruieren, sondern können es im Clipboard von Requester zu Requester übertragen.
Enthält der Zwischenbuffer ein Objekt, wird der jeweilige Menüeintrag in der normalen Menüfarbe angezeigt und kann angewählt werden. Andernfalls ist der Menüeintrag in blauer Schrift dargestellt und das Clipboard des jeweiligen Objekts enthält keinen Eintrag.

Zurück zur Objektauswahl

Ist der Requester fertig konstruiert, gelangt man über den letzten Menüeintrag des Menüs wieder zurück zur Objektauswahl. Dies kann ebenfalls durch Drücken der ESC-Taste erreicht werden.

Requester vergrößern

Der Requester kann stufenlos auf die komplette Screenbreite ausgedehnt werden. Sie müssen dazu den Mauspfeil auf die rechte untere Ecke innerhalb des Requesters bewegen, den linken Mausknopf drücken und gedrückt halten.
In diesem Zustand folgt der Maus ein Rahmenrechteck, das die neue Größe des Requesters beschreibt. Erst nach Loslassen der Maustaste wird der Requester auch tatsächlich vergrößert.

Requester Texte bearbeiten

Requestertexte sind normale Textfelder, die in unterschiedlichen Zeichensätzen dargestellt werden können. Sie haben lediglich informative Funktion. In Ihrem Programm melden Requestertexte keine Nachricht zurück, wenn sie mit der Maus angeklickt werden. Ein invers dargestellter Requestertext symbolisiert, daß das aktuelle Objekt ein Requestertext ist und Sie Zugriff auf das Text Popup-Menü haben.

Positionieren von Requestertexten

Wird ein Requestertext mit der Maus angeklickt und der Mausknopf gedrückt gehalten, können Sie den Text innerhalb der Requestergrenzen frei bewegen. Beim Loslassen des linken Mausknopfes bleibt der Text an der aktuellen Position stehen. Im Screentitel wird dann die Koordinate der linken oberen Ecke des Textes relativ zur linken oberen Ecke des Requesters angezeigt. Über die Cursortasten ist es möglich, den Text pixelweise zu verschieben. Beim Verschieben mit Maus oder Cursortasten wird
nicht der ganze Text, sondern ein Rahmen, der den Ausmaßen des Textes entspricht, verschoben. Erst nach Loslassen der Taste wird der Text nachgezogen.

Texte verändern

Dies geschieht über den Text-Requester, der sich öffnet, wenn Sie aus dem Popup-Menü den Menüpunkt „Editieren“ auswählen. Wie der Text-Requester bedient wird, ist im Kapitel „Die Programm Requester“ beschrieben. Ein Doppelklick auf einen Requestertext öffnet ebenfalls den Text-Requester.

Texte löschen

Je nach Einstellung des MakeAPP-Setup-Requesters wird bei Anwahl des Menüpunktes „Löschen" eine Sicherheitsabfrage gemacht, bevor der Requestertext gelöscht wird. Soll der Text in das Clipboard gelöscht werden, so daß er nicht mehr im Requester angezeigt wird, muß dies ebenfalls im Setup-Requester eingestellt sein.

Texte kopieren

Sollen mehrere gleiche Texte in einem Requester positioniert werden, kann die Funktion „Kopieren" aus dem Menü gewählt werden. Der kopierte Text wird dabei in die obere linke Ecke des Requesters gestellt und kann mit der Maus auf die gewünschte Position gebracht werden. Eine andere Möglichkeit Texte zu kopieren besteht darun, eine der Shift Tasten zu drücken, während ein Text angeklickt und verschoben wird. Der Text hängt dann direkt am Mauszeiger und muß nicht, wie bei der Kopierfunktion aus dem Menü, erneut angeklickt werden.

Texte ins Clipboard kopieren

Möchten Sie den Text zwischenspeichern, um ihn in andere Requester zu übertragen, besteht die Möglichkeit, ihn in das Text-Clipboard zu kopieren. Wählen Sie den Menüeintrag „In Clipboard", wird der aktuelle Text (invers) ins Clipboard kopiert. Im Requester Popup-Menü ist dann der Menüeintrag für das Text-Clipboard anwählbar.

Texte benennen

Die Namensvergabe an Texte ist wichtig für Ihr späteres Programm.
Dazu erscheint der Namens-Requester, der auch beim Neuanlegen von Requestern oder Menüleisten angezeigt wird. Der Name eines jeden Textes erscheint auch im Screentitel, wenn Sie den Requestertext mit der Maus angeklickt haben. Im Programm werden die Namen zur Makrodeklaration benutzt, die dann direkt in die MakeAPP-Funktionen der Library eingesetzt werden können.

Texte zentrieren

Die beiden letzten Einträge des Text-Popup-Menüs enthalten die Funktionen zur Zentrierung der Texte innerhalb des Requesters. Über den Eintrag „Text X Zentr." wird der Text horizontal und über „Text Y Zentr." vertikal zentriert.

Gadgets bearbeiten

Gadgets sind Requesterbereiche, auf deren Anselektierung hin in Ihrem späteren Programm eine Nachricht übermittelt wird. Von diesen Gadgets gibt es drei Arten, die auch beim Konstruieren verwendet werden können.

Boolean Gadgets:

Werden Boolean Gadgets angeklickt, erhält man je nach eingestellten Flags entweder eine Nachricht, wenn sie angeklickt oder losgelassen werden.

String Gadgets

Dies sind String Eingabefelder, deren Eingabestring in der Länge frei bestimmbar ist.

Proportional Gadgets

Dies sind Slider-Objekte wie sie zum Beispiel im Workbench-Fenster eingesetzt werden, um deren Inhalt zu scrollen.

Die jeweiligen Gadgets können Sie aus dem Requester-Popup-Menü auswählen. Ein aktuelles Gadget wird invers dargestllt. In der Titelzeile des Screens stehen Name und Gadgetart, gefolgt von Startkoordinaten und Ausmaßen des Gadgets. Wird ein Gadget mit der Maus angeklickt, so hat man Zugriff auf das Gadget-Popup-Menü, das die Funktionen zur Bearbeitung des Gadgets enthält. Gadgets positionieren und vergrößern Die Positionierung von Gadgets geschieht in der gleichen Weise wie bei Requestertexten. Ein Gadget kann aber außerdem noch in der Größe verändert werden. Klickt man auf die linke untere Ecke des Gadgets, wird ein Rahmen um das Gadget gezeichnet, der dann, wenn der Mausknopf gedrückt gehalten wird, vergrößert oder verkleinert werden kann. Der Border des Gadgets wird nicht automatisch auf die neuen Ausmaße eingestellt. Über die Shift-Cursor Tasten können Sie ein Gadget auch pixelweise vergrößern oder verkleinern.

Gadgettexte positionieren und verändern

Jedes Gadget, egal welcher Art, enthält einen eingenen Text, der auch beim Verschieben von Gadgets mitverschoben wird. Der Text steht relativ zur linken oberen Ecke des Gadgets. Der Gadgettext kann nur über die Tastatur verschoben werden. Durch Drücken einer der beiden Alernate-Tasten erscheint ein Rahmen um den Gadgettext des aktuellen Gadgets. Über die Cursortasten können Sie dann den Text verschieben. Zur Eingabe eines neuen Textes genügt ein Doppelklick auf ein Gadget und es erscheint der Text-Requester, in dem der Gadgettext verändert werden kann. Dies können Sie auch in der Menüleiste über den Menüpunkt „Edit Text" erreichen.

Gadget-Flags einstellen

In dem Requester, der erscheint, wenn der Menüpunkt „Edit Flags" angewählt wird, können Sie einstellen, wie das jeweilige Gadget in Ihrem Programm selektiert werden und reagieren soll. Je nach Gadgetart ist zusätzlich noch das Gadget „Stringinfo" oder „Propinfo" im Gadget-Flag-Requester anklickbar.

Flags für Boolean Gadgets

Selektierungs-Flags:

GADGHNONE
Das Gadget wird bei Klick mit der Maus nicht selektiert.

GADGHCOMP
Das Gadget wird bei Klick mit der Maus invertiert angezeigt.

GADGHBOX
Das Gadget wird bei Klick mit der Maus mit einem Rahmen selektiert.

GADGHIMAGE
Bei Anwahl des Gadgets soll ein anderes Image als das normal angzeigte erscheinen.

SELECTED
Das Gadget soll bereits wenn der Requester angezeigt wird selektiert sein.

GADGDISABLED
Das Gadget ist nicht anwählbar. Zur Kennzeichnung ist es mit einem Raster überzogen.

GADGIMAGE
Wählen Sie dieses Flag aus, wenn das Gadget anstelle einer Normalen Umrahmung eine Grafik anzeigen soll. Die Beschreibung über Grafiken in Gadgets finden Sie im Abschnitt „Images in Gadgets".

Activation-Flags:

RELVERIFY
Man erhält die Nachricht über die Anwahl des Gadgets bereits wenn es angeklickt wird. (GADGETDOWN)

GADGIMMEDIATE
Bei Loslassen des Gadgets wird die Nachricht über die Anwahl des Gadgets gesendet. (GADGETUP)

ENDGADGET
Der Requester soll geschlossen werden, wenn das Gadget angeklickt wird.

FOLLOWMOUSE
Während das Gadget angeklickt wird, sollen Mauskoordinaten empfangen werden können.

TOGGLESELECT
Wird das Gadget angewählt erscheint es solange in der gewählten Selektierung bis es erneut angewählt wird.Sollen die Flags für ein Stringgadget eingestellt werden, ist im Flag-Requester das Gadget Stringinfo aktiv und kann mit der Maus angewählt werden. Bei Anwahl erscheint, über dem Flag-Requester, ein weiterer Requester in dem die speziellen Stringgadget-Flags eingestellt werden können.

String-Gadget-Flags:

STRINGCENTER
Der eingegeben Text wird im Gadget zentriet angezeigt.

STRINGRIGHT
Der eingegebene Text wird rechtsbündig im Gadget angezeigt.

LONGINT
Es werden nur Zifferneingaben angenommen.
Außerdem wird in diesem Requester festgelegt, wieviel Buchstabe maximal in das Eingabefeld des Stringgadgets eingegeben werden dürfen.

Für Proportional-Gadgets enthält der Requester ein weiters Gadget, das erst aktiv ist, wenn die Flags für ein Proportional-Gadget eingestellt werden. Wie bei String-Gadgets erscheint ebenfalls ein weiterer Requester in dem die speziellen Flags für Proportional-Gadgets eingestellt werden können.

Poroportional-Gadget-Flags:

AUTOKNOB
Ist immer angewählt und kann nicht verändert werden.
FREEHORIZ
Der Slider des Gadgets soll horizontal verschiebbar sein.
FREEVERT
Der Slider des Gadgets soll vertikal verschiebbar sein.
PROPBORDERLESS
Das Gadget soll ohne Rahmen gezeichnet werden.

Die vertikale und horizontale Größe des Slider kann in den String-Eingabefeldern eingestellt werden.

Gadgets kopieren und löschen

Beim Kopieren eines Gadgets mit der Menüleistenfunktion „Kopieren" wird das Kopierresultat in der linken oberen Ecke angezeigt. Einfacher ist dies über die Tastatur. Man hält die Shift Taste gedrückt, klickt auf ein konstruiertes Gadget, das Gadget wird kopiert und es kann mit der Maus auf eine andere Position gebracht werden. Um Gadgets zu löschen muß der Menüpunkt „Löschen" im Gadget-Popup-Menü angewählt werden. Wie bei Requestertexten findet auch hier bei entsprechender Einstellung im Setup-Requester eine Sicherheitsabfrage statt.

Gadgets benennen

In den Namens-Requester kann der Name des Gadgets eingegeben werden. Der Name erscheint dann in der Screentitelzeile hinter der Gadgetartbezeichnung. Die Namen aller Objekte, seien es Gadget, Texte oder Menüs, werden vom MakeAPP in der Datei mit der Endung „.LST" abgelegt. Diese wird in Ihrem Programm benötigt, um die konstruierten Objekte einfacher über die MakeAPP-Funktionen der Library verändern zu können.

Gadgets ins Clipboard kopieren

Wählen Sie den Menüpunkt „In Clipboard" aus, wird das aktuelle Gadget in das, je nach Gadgetart vorgesehene, Clipboard kopiert. Das Gadget bleibt dabei weiterhin auf dem Bildschirm sichtbar. Im Requester-Popup-Menü ist dann der jeweilige Eintrag des Gadget-Clipboards in der normalen Menüfarbe dargestellt und kann angewählt werden.

Gadgetgrößen anpassen

Diese Funktion erleichtert Ihnen die Anpassung eines Gadgets an die Ausmaße des Gadgettextes. Hierbei wird die Gadgetstartkoordinate der Textkoordinate gleichgesetzt und das Gadget vergrößert oder verkleinert. Die Umrahmung der Gadgets wird bei dieser Funktion nicht verändert, so daß diese eventuell nachgearbeitet werden muß.

Füllmusterauswahl für Boolean Gadgets

Eine sicher ungewöhnliche Gestaltungsmöglichkeit für Gadgets sind Füllmuster. Den Requester für die Füllmusterauswahl können Sie durch Anwahl des Menüpunktes „Füllmuster" anzeigen lassen.

Der Requester enthält 36 Gadgets, die mit Füllmustern ausgefüllt sind. Wählen Sie eines der 36 Gadgets aus, erscheint dieses in der aktuellen Zeichenfarbe, unterlegt mit der Hintergrundfarbe des Requesters. Mit den Farb- und Zeichenmodi Gadgets können Sie die Darstellungsart des Füllmusters veränderen. Ist einmal ein Füllmuster ausgewählt, kann dieses nicht mehr rückgängig gemacht werden.

Gadget- und Gadgettextzentrierung

Die Menüpunkte „X-Zentrierung" und „Y-Zentrierung" können ausgewählt werden, um das Gadget im Requester zu vermitteln. Mit den Menüpunkten „Text X Zentr." und „Text Y Zentr" wird der Gadgettext innerhalb der Ausmaße des Gadgets zentriert.

Gadget Border verändern

Border sind Linienzüge, die eine beliebige Form annehmen können. Sie gehören fest zu einem Gadget und begrenzen meist die Ausmaße derer. Im Border-Editor, der über den Menüpunkt „Gadget Border" erreichbar ist, können sie die Border frei gestalten. Dazu wird ein neuer Requester geöffnet und der Border des aktuellen Gadgets in der Mitte des Requesters angezeigt. Mit dem Mauszeiger können Sie nun die Linienzüge neu gestalten. Jede Eckkoordinate des Borders ist mit einem Punkt gekennzeichnet. Der aktuelle Punkt ist durch einen etwas dickeren Rahmen hervorgehoben. Ein neuer Eckpunkt kann ganz leicht hinzugefügt werden, indem Sie eine der Linien, an der Stelle, an der der neue Punkt liegen soll, anklicken. Lassen Sie die Maustaste nicht los, wenn Sie einen Punkt angewählt haben, so kann er verschoben werden. Die Linie wandert dann wie ein Gummiband der Maus nach.

Im Gadget-Border-Menü sind Funktionen enthalten, die zur Gestaltung eingesetzt werden können.

Neue Border hinzufügen

Über den ersten Menüeintrag werden neue Border hinzugefügt. Es können maximal fünf Border an ein Gadget angehängt werden. Diese fünf sind unabhängig voneinander gestaltbar. Sie können andere Formen und andere Farben haben. Welcher Border editiert werden soll, ist aus den fünf nachfolgenden Menüeinträgen wählbar. Der aktuelle Border ist im Menü mit einem Haken gekennzeichnet.

Border Clipboard

Ein Gadget-Border kann auch als normaler Requester-Border eingesetzt werden. Dazu muß er in das Clipboard kopiert werden, wodurch der aktuelle Border keineswegs vom Bildschirm verschwindet. Verläßt man nun den Gadget-Border-Editor kann der im Clipboard enthaltene Border aus dem Requester-Popup-Menü zum Requester hinzugefügt und angezeigt werden. Wie Requester-Border gehandhabt werden, erfahren Sie im nächsten Abschnitt. Sie können auch Requester-Border, die in das Clipboard kopiert wurden, mit dem Menüpunkt „Aus Clipboard" zu einem Gadget hinzufügen. Der Border aus dem Clipboard wird dabei anstelle des aktuellen Borders eingesetzt.

Border und Borderpunkte löschen

Soll der aktuelle Border gelöscht werden, müssen Sie aus dem Menü den Eintrag „Akt. Border löschen" auswählen. Hierbei findet wieder, je nach Einstellung des Setup-Requesters eine Sicherheitsabfrage statt. Die Eckkoordinaten der Linien werden als Borderpunkte bezeichnet. Der aktuelle Punkt ist durch eine dickere Umrahmung gekennzeichnet. Werden solche Punkte gelöscht, wird keine Sicherheitsabfrage gemacht.

Gestaltungshilfmittel

Im Editor-Requester ist immer ein Fadenkreuz angezeigt, daß den Requester in vier gleiche Teile unterteilt. Weitere Hilfslinien können mit dem Menüpunkt „Hilfslinien" angezeigt oder auch wieder entfernt werden. Im Editor können Sie neben dem aktuellen Border auch alle anderen bereits konstruierten Border des Gadgets anzeigen lassen.

Symmetrisch editieren

Über den Menüpunkt „Symmetrisch" kann in den Symmetrie-Modus umgeschaltet werden. In diesem Modus können Sie den Border nur im linken oberen Quadrat verändern. Die Inhalte der anderen Quadrate werden dann entsprechend gespiegelt. In diesem Modus können sehr einfach runde Border konstruiert werden.

Tastatursteuerung

Viele Funktionen können Sie auch über die Tastatur erreichen. Mit den Cursortasten kann der aktuelle Punkt pixelweise verschoben werden. Über die „Pg Up"- und „Pg Dn"-Tasten der Zehnertastatur kann der aktuelle Punkt auf den nächsten oder vorherigen gesetzt werden. „Delete" löscht den aktuellen Punkt ohne Sicherheitsabfrage. Mit der ESC-Taste gelangen Sie in den Requester-Editor zurück.

Images in Gadgets

Images sind Grafiken, die innerhalb von Boolean Gadgets angezeigt werden können. Im MakeAPP ist es möglich, normale IFF-Grafiken in solchen Images zu konvertieren. Im Gadget-Flag-Requester muß dazu das Flag GADGIMAGE angewählt werden. Die Menüeinträge „Gadget Border" und „Select Border" des Gadget- Popup-Menüs werden dann in „Gadget Image" und „Select Image" umbenannt.
Intuition verbietet es, Gadget-Border mit Images zu kombinieren. Anstelle des Border-Editors erscheint der Image-Editor. Dabei wird der komplette Inhalt des Requesters angezeigt. Es kann allerdings nur noch das aktuelle Gadget verschoben und vergrößert werden. Über das Image-Popup-Menü lassen sich Iff-Grafiken laden, auswählen und Auschnitte festlegen. Bitte achten Sie darauf, daß Ihre Images nicht zu groß werden. Generell ist die Größe des Images auf die Größe des gewählten Bildes begrenzt, was aber meist nicht weniger als 30 KByte Speicherplatz für das Image bedeutet. Man wird leicht verleitet, zu große und zu viele Grafiken in seine Requester zu packen, was dann zu unangenehmen Folgen führen kann. Vergewissern Sie sich deshalb immer, ob der benötigte Speicherplatz für die Ressource-Datei nicht zu groß wird.

Laden, Auswählen und Löschen von Grafiken

Beim Laden wird der Datei-Auswahl-Requester angezeigt, aus dem Sie Ihr Bild auswählen können. Konnte ein Bild erfolgreich geladen werden, wird ein Ausschnitt des Bildes, je nach Größe des Gadgets angezeigt. Anderenfalls steht die entsprechende Fehlermeldung im Screentitel. Es können maximal acht Bilder im Speicher gehalten werden. Um bereits geladene Bilder auswählen oder aus dem Speicher löschen zu können, wird der Auswahl-Requester geöffnet, der auch zur Fontauswahl im Text-Requester genutzt wird.

Ausschnitt festlegen

Wird ein Iffbild ausgewählt, so ist der angezeigte Ausschnitt der linke obere Teil des Bildes. Das Gadget kann über die Tastatur oder mit der Maus vergrößert werden, wobei automatisch auch der Ausschnitt vergrößert wird. Mit den Pfeil-Gadgets, die im Popup-Menü stehen, können Sie das Bild im Gadget scrollen. Öffnen Sie dazu das Menü und klicken eines der Gadgets im Menü mit der linken Maustaste an. Der Ausschnitt kann auch mit den Ctrl-Cursortasten der Tastatur gescrollt werden.

Farben übernehmen und Image konvertieren

Da die voreingestellten Requesterfarben meist nicht zu einem Bild passen, können Sie mit dem Menüpunkt „Farben übernehmen" die Farben, die zu dem Bild abgespeichert wurden, als neue Requester-Farbpalette übernehmen. Mit der ESC-Taste oder Anwahl des Menüpunktes „Zurück" wird der Grafikausschnitt in ein Image konvertiert. Bei großen Grafiken kann dies eine kurze Zeit dauern.

Requester-Border-Editor

Requester-Border sind ähnlich den Gadget-Bordern, können jedoch als einzelne Objekte im Requester frei bewegt werden. Neue Requester-Border werden über den Menüpunkt „Border" im Requester-Popup-Menü zum Requester hinzugefügt. Bei einem aktuellen Border sind die Eckpunkte des Borders, wie im Gadget-Border-Editor mit kleinen Rahmen markiert. Diese Rahmen sind die Griffpunkte für die Maus. Da im Requester-Border-Editor, anders als im Gadget-Border-Editor, kein eigener Editor-Requester geöffnet wird, kann der Requester-Border über andere, bereits im Requester vorhandene

Objekte verschoben und verändert werden. Im Border-Popup-Menü können Sie die Funktionen zum Kopieren, Löschen, Zentrieren und Editieren erreichen.

Requester-Border positionieren

Bei Requester-Bordern genügt es nicht, eine der Linien anzuklicken, um den Border zu verschieben. Es muß unbedingt eine der Ecken des Border getroffen werden. Sind mehrere Border im Requester angezeigt und Sie möchten von einem zum anderen Border wechseln, muß dies ebenfalls durch Anklicken einer der Bordereckpunkte geschehen. Hat man einen Border angewählt, kann dieser auch über die Cursortastatur im Requester verschoben werden.

Border löschen und kopieren

Der Border kann genau wie alle anderen Objekte wieder aus dem Speicher gelöscht werden. Dabei werden Sie je nach Einstellung des Setup-Requesters gefragt, ob dies beabsichtigt ist. Ein Border kann auf zwei Arten kopiert werden. Entweder wählen Sie den Menüpunkt „Kopieren" an, wobei die Startkoordinate des neuen Borders in der linken oberen Ecke des Requesters liegt, oder Sie kopieren ihn, indem Sie eine der Shift-Tasten gedrückt halten, den Eckpunkt des Borders anklicken und den neuen Border auf die gewünschte Position bringen.

Border zentrieren und Farbe einstellen

Mit den Menüpunkten „X-Zentrierung" und „Y-Zentrierung" können Sie den Border horizontal und vertikal im Requester zentrieren. Die Borderfarbe und -zeichenmodi sind im Farb-Requester einstellbar.

Border editieren

Um einen Border verändern zu können, muß in den Editiermodus umgeschaltet werden. In diesem Modus könne Sie den Border nicht mehr verschieben. Klickt man mit der Maus auf einen Eckpunkt oder eine Linie und hält die Maus gedrückt, haftet der angeklickte Borderteil solange an der Maus, bis die Maustaste wieder losgelassen wird.

Im Symmetriemodus, der über das kleine Popup-Menü des Editors eingeschaltet werden kann, ist es möglich, runde Border oder exakte Rechtecke zu konstruieren. Dabei wird der Border in die Mitte des Requesters gestellt und das Fadenkreuz, wie es aus dem Gadget-Border-Editor bekannt ist, angezeigt. Sie können den Border dann nur noch im linken oberen Quadrat des Requesters verändern. Alle anderen

Quadrate werden entsprehend gespiegelt. Die Cursortasten können Sie benutzen, um den aktuellen Punkt, der dick umrandet markiert ist, zu verschieben. Mit den „Pg Up"- und „Pg Dn"-Tasten der Zehnertastatur wird der aktuelle Punkt auf den nächsten oder vorherigen Eckpunkt gestellt. Hat man den Border fertig konstruiert, kann der Border-Editor verlassen werden, indem Sie im Menü den Eintrag „Zurück" anwählen oder mit der Maus außerhalb des Requesters klicken.

Menüleisten

Der Menü-Editor von MakeAPP wird ebenfalls auf einem eigenen Screen angezeigt. Um ihn aufzurufen, genügt ein zweifacher Klicken auf ein Menü-Image im Objektauswahlfenster. Neue Menüleisten können hinzugefügt werden, indem Sie in der Objektauswahl das Menü „Objektbearbeitung" anwählen und die rechte Maustaste über dem Eintrag „Menüleiste hinzufügen" loslassen.

Unter der Titelzeile des Menü-Editorscreens ist eine weitere Menütitelzeile angebracht. In ihr werden die Titel der Menüleisten sichtbar. Der Editor simuliert während des Konstruierens eine echte Amiga-Menüleiste, um diese immer komplett im Screen anzeigen zu können. Es wäre Unsinn, in einem komfortablen Editor ständig die rechte Maustaste gedrückt halten zu müssen.

Jedem Menübestandteil können Sie mit der Maus anklicken und somit aktualisieren. In der Screentitelzeile wird dann jeweils angezeigt, um welchen Bestandteil es sich handelt, welchen Namen er hat und auf welcher Koordinate, relativ zum Mutterobjekt, er angebracht wurde.

Menütitel hinzufügen

Ist der Editorscreen leer angezeigt, müssen zuerst Menütitel in der Menüleiste angebracht werden. Dazu enthält das Popup-Menü „Menü" den Eintrag „Titel". Dieser kann sooft angewählt werden, wie Titel in der Menüleiste vorhanden sein sollen. Die Menütitel werden hintereinander in die Menüleiste gestellt.

Menutiteltexte ändern

Beim Einfügen eines neuen Menütitels wird der Text „Titel" in der Menüzeile angezeigt. Um diesen zu ändern, muß der Titel mit der Maus angeklickt werden. Er erscheint dann invers. Dabei wird automatisch ein neues Popup-Menü zugänglich, das die Funktionen zum Ändern des Textes und Hinzufügen von Einträgen beinhaltet. Wählen Sie aus diesem Menü den Eintrag „Menü Text" an, wird der Namens-

Requester geöffnet, der dann dazu benutzt werden kann, dem Menü einen neuen Titel zu geben. Für Menütitel können keine Fonts oder Farben eingestellt werden.

Anbringen des Pull-Down-Menüs

Ein Pull-Down-Menü besteht aus Menüeinträgen und beim Amiga zusätzlich noch aus Untereinträgen. Wieviele Einträge und Untereinträge in einem Menü enthalten sein sollen, ist nur durch die Größe des Bildschirms begrenzt. Um neue Menüeinträge an einem Titel hinzufügen zu können, muß zuerst der entsprechende Titel mit der Maus angeklickt werden. Wurden bereits Einträge an einen Menütitel angehängt, wird die Liste der Einträge, wie in einem normalen Pull-Down-Menü, aufgeklappt. In diesem Zustand ist nur der Menütitel invers dargestellt. Mit dem Menüpunkt „Eintrag" aus dem Titel-Popup-Menü können Sie neue Einträge an den Menütitel anhängen. Zu jedem Menüeintrag kann ein Untermenü angehängt werden. Dieses erscheint dann in der echten Menüleiste, wenn der entsprechende Eintrag mit der Maus angewählt wird. Im Menü-Editor muß dazu der Eintrag mit der Maus angeklickt werden. Dieser wird dann invers angezeigt und ein neues Popup-Menü ist zugänglich. Mit dem Menüpunkt „Untereinträge" können Sie Untereinträge an den aktuellen Eintrag anhängen. Diese Untereinträge werden jedesmal angezeigt, wenn Sie im Editor den entsprechenden Eintrag anklicken.

Verschieben von Titeln, Einträgen und Untereinträgen

Die Menütitel können horizontal innerhalb der Menüleiste verschoben werden. Das Pull-Down-Menü des Menütitels wird dabei mitverschoben und an der neuen Position angezeigt. Einträge und Untereinträge können Sie wie Gadgets handhaben. Werden Sie angeklickt, können Sie, solange der Mausknopf gedrückt bleibt, innerhalb des Screens verschieben. Die pixelweise Feinpositonierung ist mit den Cursortasten möglich.

Vergrößern von Einträgen und Untereinträgen.

Der Selektierungbereich eines Menüeintrags ist beliebig vergrößerbar. Bei Klicken auf den rechten unteren Bereich der Selektierung erscheint ein Rahmen in der Größe des Selektierungsbereichs. Bewegen Sie jetzt die Maus bei gedrücker linker Maustaste, so haftet der Rahmen, der die neue Größe des Selektierungbereichs markiert, an dem Mauzeiger. Gleiches gilt für Untereinträge. Mit den Alt-Cursor-Tasten können Sie die Größe des aktuellen Eintrags oder Untereintrags pixelweise Verändern.

Texte verschieben und ändern

Texte von Menüeinträgen können generell nur innerhalb des Selektierungsbereichs und nur mit der Tastatur verschoben werden. Beim Drücken einer der Alt-Tasten wird ein Rahmen um den Text des aktuellen Menüeintrages gezeichnet. Diesen Rahmen können Sie mit den Cursortasten auf die gewünschte Position, fahren. Erst nach Loslassen der Taste wird der Text des Eintrags an die neue Position gestellt. Durch einen Doppelklick auf einen Menüteil, egal ob Titel, Eintrag oder Untereintrag, wird ein Requester geöffnet, in dem ein neuer Text eingegeben werden kann. Bei Menütiteln ist dies der Namens-Requester, der bereits beschrieben wurde. Bei Einträgen und Untereinträgen wird der Text-Requester angezeigt, in dem Sie auch Zeichensätze, Schriftfarbe und -zeichenmodi wählen können.

Menü-Flags

Der Requester für die Einstellung der Menüflags wird nur in den Pop-Up-Menüs für Einträge und Untereinträge zugänglich. Wählen Sie den Menüpunkt „Menü Flags" im Pop-Up an, wird ein Requester geöffnet in dem folgende Flags eingestellt werden können:

Selektierungs-Flags:

HIGHIMAGE
Sobald sich der Mauszeiger bei gedrückter rechter Maustaste über dem Menüeintrag befindet, wird ein Selekt-Text oder Selekt-Image angezeigt. Ist dieses Flag eingeschaltet, ist der Eintrag „Select Text" im Pop-Up-Menü anwählbar.

HIGHCOMP
Der Menüeintrag wird invertiert, sobald man ihn , bei gedrückter rechter Maustaste, mit der Maus anwält.

HIGHBOX
Befindet sich der Mauszeiger bei gedrückter rechter Maustaste über den Eintrag, wird um diesem ein Rahmen gezeichnet.

CHECKED
Vor dem Text des Menueintrages soll eine Checkmark stehen. Dies ist nur in verbindung mit dem Activation-Flags CHECKIT möglich.

Activation-Flags:

CHECKIT
Der Eintrag soll mit einer Checkmark versehen werden können.

ITEMTEXT
Dieses Flag ist immer angewählt und kann in dieser Version des MakeAPP nicht verändert werden. In einer späteren Version wird es möglich sein in Menüleisten ebenfalls Images anzeigen zu lassen.

COMMSEQ
Wählen Sie dieses Flag, wenn der Menüeintrag alternativ auch durch die Tastatur ausgewählt werden können soll. Das ensprechende Ascii-Zeichen für den „Shortcut" kann im String-Gadget unterhalb der Flag-Gadgets eingegeben werden. Normalerweise wird hinter dem Menütext ein Aimga-A Symbol angezeigt, was in der Simuation der Menuleiste nicht gemacht wird. Möchten Sie sich die Menüleiste ansehen, wie sie später in Ihrem Programm aussehen wird, wählen Sie den Menüpunkt „Menü Testen" aus den Pop-Up-Menüs.

MENUTOGGLE
Setzen Sie dieses Flag, wird bei Auswahl des Menüeintrages dieser mit einer Checkmark versehen und bei erneuter Auswahl wieder entfernt. Das Flag kann nur in kombination mit dem Flag CHECKIT eingesetzt werden.

ITEMANABLED
Dieses Flag ist gesetzt, wenn der Einträge aus der Menüleiste angewählt werden kann. Ist er nicht gesetzt, wird der Menueintrag mit einem Raster unterlegt und kann nicht angewählt werden.

Menü testen

Um die gemachten Einstellungen in der Original-Intuition-Menüleiste sehen zu können, besteht die Möglichkeit, das Menü zu testen. Jedes Popup-Menü des Menü-Editors enthält dafür den Menüpunkt „Test Menü". Hat man den Menüpunkt angewählt, so kann das Menü wie übliche Menüs angezeigt und behandelt werden. Ein Klicken auf die linke Maustaste genügt, um die Menüleiste weiter verändern zu können.

Menüs justieren

Die Inhalte von aufgeklappten Intuition-Menüs stehen meist in einer Linie untereinander. Beim Anklicken von Menüteilen im Menü-Editor passiert es aber leicht, daß das angewählte Teil dabei verschoben wird und nur mit sehr viel Geduld wieder auf die alte Position zu bringen ist. Im Justiermodus wird nicht nur das verhindert, sondern auch alle Menüeinträge und Untereinträge automatisch in einer Linie untereinander gestellt. Dabei ist es nicht mehr möglich, Einträge frei zu positionieren. Die Selektierungsbereiche von neu hinzugefügten Einträgen werden an die Größe aller anderen, in diesem Pull-Down-Menü vorhandenen Menüteile, angepaßt. Um immer in diesem Modus zu arbeiten, kann „Justieren" im „MENÜ" Popup global eingeschaltet werden. Der jeweilige Eintrag wird dann mit einem Haken versehen.

Soll die automatische Justierung nicht immer gemacht werden, können Sie aus den Popup-Menüs für Einträge und Untereinträge den Menüpunkt „Justieren" anwählen. Dabei wird das Menü, unter Berücksichtigung der Textkoordinate und des Selektierungsbereiches, an das gerade aktuelle Menüteil angepaßt.

Tastaturbelegungstabelle

Für alle Editiermodi

ESC	Zurück.

Im Requester-Editor

Cursortasten:	Pixelweise Steuerung von Texten, Gadgets und Bordern.
Shift-Maus:	Kopieren von Texten, Gadgets und Bordern.
Shift-Cursor-Tasten:	Pixelweises Vergrößern und Verkleinern von Gadgets.
Alt-Cursor:	Verschieben von Gadgettexten.

Bei gedrückter **Taste B**, kann man mit der Maus einen Border anwählen, der sich unter einem Gadget befindet.
Bei gedrückter **Taste T**, kann man mit der Maus einen Requestertext anwählen, der sich unter einem Gadget befindet.

Im Gadaget- und Requester-Border-Editor

Cursortasten:	Pixelweises Verschieben des aktuellen Punkts.
Del:	Löschen des aktuellen Punkts.
Pg Up und Pg Dn:	Setzten des aktuellen Punkts auf den nächsten oder vorherigen Eckpunkt.

Im Image Editor

Cursortasten:	Pixelweises Verschieben des Gadgets.
Shift-Cursor-Tasten:	Pixelweises Vergrößern und Verkleinern des Gadgets.
Alt-Cursor:	Verschieben des Gadgettextes.
Ctrl-Cursor:	Pixelweises Scrollen des Bildes. Im Menüleisten-Editor
Cursortasten:	Pixelweises Verschieben von Titeln, Einträgen und Untereinträgen.
Shift-Cursor-Tasten:	Vergrößern von Einträgen und Untereinträgen.
Alt-Cursor-Tasten:	Pixelweises Verschieben von Menütexten Maustasten

Doppelklick: Für alle Textobjekte (Requestertexte, Gadgets, Menütitel, Menüeinträge und Untereinträge) wird der Text-Requester geöffnet.

Shift-Mausklick mit linker Maustaste kopiert Objekte aller Art.

Die Ressource-Bibliothek

MAXON
computer

Nutzungsrecht

Mit den Bibliotheksfunktionen der Library läßt sich sehr viel einfacher programmieren, als wenn man Sourcecode generiert und diesen weiterbearbeitet. Desweiteren hat man sich um die Verwaltung der im Source vorhandenen Oberfläche selbst zu bemühen. Änderungen an der Benutzeroberfläche sind wesentlich einfacher. Es muß nicht erneut Sourcecode generiert, umständlich in das Programm eingebunden und compiliert werden. Damit Sie den Komfort der Bibliothek uneingeschränkt auch für Ihre eigenen Programme nutzen können, sind die Bibliothek selbst (rct.library) und alle anderen im „Libraries-Verzeichnis" der Original-Diskette enthaltenen Dateien Public Domain (PD). Sie dürfen die Bibliothek und die Schnittstellen-Dateien zu Ihren Programmen dazulegen. Wir würden uns sehr freuen, wenn Sie mit einem Hinweis in Ihren Programmen oder im entsprechenden Programm-Handbuch vermerken, daß es unter Zuhilfenahme von MaxonMakeAPP und der RCT-Library entstanden ist.

Inhaltsverzeichnis

Nutzungsrecht 2
Einleitung 5
Der erste Kontakt 8
Die beiden wichtigsten Funktionen der Bibliothek 9
Die Grundlagen allen Schaffens - Screens und Fenster 13
Die Library-Requester 33
Das Screen-Mode Auswahlfenster 44
Lokalisierung und die Auswahlmöglichkeit 47
Menü-Funktionen 72
Kompatibilitätsfunktionen 78
Anhang 85
Anpassung von alten Programmen 87

Index 89

Einleitung

Ein großer Vorteil des Amiga Betriebssystems ist die problemlose Erweiterung durch sogenannte „Shared-Libraries". Das englische Wort „share" bedeutet in der deutschen Übersetzung „ Anteil, teilhaben..." auf das Amiga-Betriebssystem bezogen, bedeutet das, daß mehrere Programme gleichzeitig auf die Funktionen der Bibliothek zugreifen können. Die Funktions-Bibliothek selbst ist dabei allerdings nur einmal geladen und belegt auch nur einmal Speicher. Ein weiterer Vorteil solcher „Shared-Libraries" ist die Speicherfreigabe, wenn sie nicht mehr benötigt werden. Darum kümmert sich das Betriebssystem selbst. Sobald der Speicher im System knapp wird, prüft es nach, welche verfügbaren Bibliotheken von Progammen genutzt werden und welche nicht. Der belegte Speicher der ungenutzten Bibliotheken wird dann freigegeben.
Die Bibliotheks-Funktionen können mit jeder Programmiersprache genutzt werden, die es erlaubt, Betriebssystemfunktionen direkt aufzurufen. Es gibt nur sehr wenige Programmiersprachen, die diese Möglichkeit nicht oder nur sehr eingeschränkt bieten.
Folgende Programmiersprachen kommen problemlos damit zurecht:

Assembler
C/C++-Compiler
Basic-Dialekte wie AMIGA-Basic/GFA-Basic/HiSoft-Basic
Pascal
Modula
Oberon
Cluster

Die RCT.LIBRARY

Bei der Ressource-Bibliothek (`rct.library`) handelt es sich ebenfalls um eine „Shared-Library". Entwickelt wurde sie ursprünglich für das Resource-Construction-Tool, welches hierdurch bereits 1988 die Möglichkeit bot, die komplette mit R.C.T. erstellte Benutzerführung ihrer eigenen Programme während der Programmlaufzeit nachzuladen und zu verwalten. Es entfällt die Compilierung und Initialisierung der Daten direkt in Ihrem Programm. Es ist jederzeit möglich, am Bildschirm Veränderungen vorzunehmen, und Sie müssen Ihr Programm nicht neu compilieren, wenn es sich nur um kleine, kosmetische Veränderungen handelt.

Fortschritt

Im Laufe der Jahre kamen weitere Funktionen hinzu, die die Bibliothek für jeden Programmierer, der das Rad nicht jedesmal erneut erfinden möchte, interessant machen. Bei dieser Gelegenheit möchte ich ganz besonders Thomas E. Wieger, Christian Spreuer und Herrn Jürgen Loh danken, die mit sehr vielen konstruktiven Anregungen an einigen Erweiterungen beteiligt waren.

Komprimiert

Es ist natürlich weiterhin Möglich die Ressource-Daten, egal ob sie vom Programm MakeAPP oder R.C.T. erzeugt werden, nachzuladen und zu verwalten. Hierbei wurde jedoch einiges verbessert. Wir entwickelten ein spezielles Komprimierverfahren, welches die Oberflächen-Datei erheblich verkleinert. Auch ist der Speicherverbrauch der Daten nach dem Einladen mit den Bibliotheksfunktionen wesentlich geringer. Das Laden mehrerer Ressource-Dateien ist nun auch kein Problem mehr.

Direkt Angebunden

Mit der Funktion **R_ResLoad** kann die Oberflächen-Datei nicht nur von Diskette oder Festplatte nachgeladen werden, sondern auch aus dem Speicher. Es besteht dadurch die Möglichkeit, die Oberflächen-Daten direkt in Ihr Programm zu übernehmen. Ein Suchen der Datei in verschiedenen Pfaden ist dadurch vermeidbar.

Lokalisiert

Beim Speichern der Dateien aus MakeAPP kann zusätzlich eine Textdatei geschrieben werden, in der alle Texte aus Intuitext-Strukturen enthalten sind. Beim Laden der Datei mit der Bibliotheksfunktion kann zusätzlich die Textdatei geladen werden, deren Texte dann automatisch wieder in die Intuitext-Strukturen übernommen werden. Es besteht dadurch die Möglichkeit, zum einen Textdateien in verschiedene Landessprachen auszugeben und zum anderen, die Tastaturkürzel-Definitionen vom Anwender frei definieren zu lassen.

Tastaturkürzel

Wie allgemein üblich, wird die Tastaturkürzeldefinition im Text des jeweiligen Gadgets selbst vorgenommen. Ein Unterstrich vor einem Zeichen (z.B. _Ja) bedeutet, daß bei der Anzeige des Textes der nachfolgende Buchstabe unterstrichen (_Ja wird zu Ja) und der ASCII-Wert des entsprechenden Zeichens als Auswahlkürzel genutzt werden soll. Ein doppelter Unterstrich (z.B. _ _Ja) bedeutet, daß kein Kürzel und damit verbunden kein Unterstrich unter den Buchstaben, sondern tatsächlich nur ein Unterstrich vor den Buchstaben gesetzt wird(_ _Ja wird zu _Ja). Die Bibliotheks-Funktionen **R_GetMsg** und **R_ReplyMsg** filtern die Nachrichten automatisch für Sie aus, so daß Sie in Ihrem Programm zum Beispiel nur auf Gadget-Nachrichten warten müssen, und diese auch bekommen obwohl die Tastatur bedient wurde.

Immer die richtige Farbdarstellung

Egal wie Sie die Farben für Ihre Programme auch definiert haben, die Umrahmungen für Gadgets, die Texte und Füllmusterfarben erscheinen immer in einer kontrastreichen Zusammenstellung, so daß der „3-D Look" erhalten bleibt.

Dateiauswahlfenster

Das Dateiauswahlfenster, das bereits in der alten Version sehr komfortabel war, konnte erneut verschnellert werden und erhielt weitere interessante Funktionen. Auch die Programmeinbindung wurde erhebelich vereinfacht. Lesen Sie mehr dazu im Abschnitt zur Funktion **R_Filer**.

Alarmfenster

Ein sehr vielseitiges Dialogfenster ist der Alarmdialog, der mit der Funktion **R_FormAlert** aufgerufen wird. Er kann zum einfachen Anzeigen von Fehlermeldungen genau so genutzt werden, wie zur Auswahl mehrer Kriterien. Die Abfrage-Gadgets sind auch mit Tastaturkürzeln belegbar.

Fontauswahlfenster

Mittlerweile besteht in sehr vielen Programmen die Möglichkeit einer Fontauswahl für die Anzeige von Texten in Fenstern und in Menüleisten. Der Anwender kann sich so den Zeichensatz einstellen, mit dem er am liebsten arbeitet. Für Sie als Software-Entwickler bedeutet dies einen erheblichen Programmieraufwand, der Ihnen mit dem Fontauswahlfenster abgenommen wird.

Auflösungenauswahlfenster

Mit fortwährenden Steigerungen der Grafikfähigkeiten erweitert sich auch die Palette der Screeneinstellungen für die Pixelauflösung und Darstellungsmodi. Das Auflösungsauswahlfenster sorgt dafür, daß in Ihren Programmen immer die auf dem jeweiligen System verfügbaren Auflösungen ausgewählt werden können.

Fontsensitiv

Nicht nur die Texte von Menüleisten, sondern auch die innerhalb von Gadgets und Dialogboxen werden in der Höhe und Breite automatisch angepaßt. Dabei wird der Screenfont des jeweiligen Screens benutzt, auf dem dargestellt wird.

Der erste Kontakt

Bevor Sie die Funktionen in Ihren Programmen nutzen können, muß die Library geöffnet werden. Dies geschieht sehr einfach, wie in dem folgenden Beispiel verdeutlicht:

```
#include "exec/exec.h"
#include "rct/rctdef.h"
#include "rct/rctpragma.h"
#include "rct/rcttagfuncs.h"

APTR RctBase;
APTR ApplAddr;

void main( void)
{
          //      Öffnen der Library
if( !( RctBase = ( APTR)OpenLibrary(  ( UBYTE *)"rct.library", 16L)))
  exit( -1);
if( !(ApplAddr = ( APTR)R_InitApplTags( TAG_DONE, 0L)))
  {
  CloseLibrary( ( struct Library *)RctBase);    // Sauber schließen!
  exit( -1);
  }
  .... // Hier geht es weiter im Programm

  // Programm beenden!
R_ExitAppl( ApplAddr);
exit( 0);
}
```

Taglisten

Sehr vielen Funktionen der Bibliothek können „Taglisten" (variable Parameterlisten) übergeben werden. Die Parameter der Funktionen sind dadurch, bei zukünftigen Neuerungen problemlos erweiterbar.
Eine Tagliste ist ein zusammenhängender Speicherblock, in den die Parameter in einem festgelegten Format geschrieben werden.

Die TagItem-Struktur:

```
struct TagItem
   {
   ULONG ti_Tag;
   ULONG ti_Data;
   };
```

Jeder Parameter wird in der oben beschriebenen Struktur angegeben.
Der Wert in „ti_Tag" identifiziert den Parameter und „ti_Data" enthält den Wert dazu.

Bei machen Einstellungen ist allerdings ein Wert nur zum An- oder Abschalten gewünschter Eigenschaften gedacht (Boolean-Tags). Sie sehen also: Mit einer solchen Liste lassen sich sehr einfach variable Parameterübergaben an Funktionen realisieren. Für spätere Erweiterungen steht nichts im Wege und alles bleibt ohne viel Aufwand weiterhin kompatibel.

Beispiel:

```
...
struct TagItem tags[10];   /* Nicht mehr als 10 Parameter (80 Byte) */
tags[0].ti_Tag = RFA_WindowFlags; /* Identifier */
tags[0].ti_Data = WINDOWCLOSE|WINDOWDEPTH;      /* Wert */
tags[1].ti_Tag = RFA_DefaultID;   /* Identifier */
tags[1].ti_Data = 1L;   /* Wert */
tags[2].ti_Tag = RFA_AlertText;   /* Identifier */
tags[2].ti_Data = "[1][Fehler bei Ausgeben der|Druckerdatei!][Nochmal
versuchen|Abbrechen]";  /* Wert */
tags[3].ti_Tag = TAG_DONE; /* Listenende Marke */
tags[3].ti_Data = 0L;
R_FormAlert( ApplAddr, tags); /* Funktionsaufruf mit Parameterliste */
```

Die beiden wichtigsten Funktionen der Bibliothek

R_InitAppl (V16)

R_InitApplTags (V16)

appladdr = R_InitAppl(taglist)
D0 A0

appladdr = R_InitApplTags(tag, ...)

APTR	appladdr;	/* Rückgabewert */
struct TagItem	*taglist;	/* Liste der Eingabeparameter */

Die Funktionen *R_InitAppl* und *R_ExitAppl* sind in der neuen Version der Bibliothek die beiden wichtigsten. *R_InitAppl* sollte unmittelbar nach dem Öffen der RCT-Library aufgerufen werden. Von ihr werden einige Systemressourcen angefordert, was bei Speicherknappheit eventuell fehlschlagen kann.
Wird ein Wert kleiner oder gleich NULL zurückgeliefert, ist ein Fehler aufgetreten.

Mit den Tag-Parametern können gewisse Voreinstellungen gemacht werden.

Die TAG-Parameter von **R_InitAppl:**

RIA_Version
Die Versionsnummer der Library, die beim Programmieren verwendet wird.

Das Format der Versionsnummer im ti_Data-Feld:
VersionsnummerRevisionsnummer

Beispiel:
Die Versionsnummer ist 16
Die Revisionsnummer ist 24

TagItem->ti_Data = 1624;

RIA_AutoRenderColor
Die Farben von Border und Texten sollen bei Aufruf durch *R_ReqDraw*, *R_ReqDrawWindow* und *R_GadDrawWindow* automatisch an die Screenfarben angepasst werden, auf dem dargestellt wird. Dazu wird die beste Farbkombination gewählt, um immer einen "OS 2.0 - 3D-Look" zu erzeugen. Unter OS 1.3 und den standardmäßig gewählten Farben erscheinen Gadgetborder auch korrekt im "3-D Look". „ti_Data" ist ein Boolean (TRUE oder FALSE).

Achtung: Die Farbanpassung funktioniert nicht automatisch bei Requesterborder. Für Gadget-Border wird der erste Border in der Highlight-Farbe und der zweite in der Shadow Farbe gezeichnet. Hat man beim Konstruieren in MakeAPP einfach nur die Borderfarben geändert, um einen tiefgelegten 3-D Eindruck zu erzeugen, wird dies bei der automatischen Anpassung nicht berücksichtigt. Es muß in dem Fall die Borderreihenfolge geändert werden.

Vorgehensweise: Im Gadget-Bordereditor kopieren Sie den ersten Border in den Zwischenspeicher, löschen diesen, erzeugen einen neuen und kopieren in den neu erzeugten Border den zuvor in den Zwischenspeicher kopierten. Dann können Sie die Farbe der Border ändern (weiß -> schwarz, schwarz -> zu weiß).

RIA_SameIDCMP
Alle Fenster, die mit den Bibliotheksfunktionen geöffnet werden, erhalten automatisch den gleichen IDCMPort. Das betrifft auch die Fenster für die Dateiauswahl, das Alertfenster, den Fontauswahldialog und das Auflösungs-Auswahlfenster. Beim Öffnen des ersten Fensters wird der Port automatisch eingerichtet und erst beim Beenden des Programmes durch **R_ExitAppl** wird dieser Port wieder geschlossen. „ti_Data" ist ein Boolean (TRUE oder FALSE).

RIA_GloblPort

Alle Fenster, die mit den Bibliotheksfunktionen geöffnet werden, erhalten ebenfalls automatisch den gleichen IDCMPort. In „ti_Data" wird die Adresse des Ports, der durch Ihr Progamm geöffnet wurde, übergeben. Die Angabe dieses Parameters überschreibt den Parameter RIA_SameIDCMP.

RIA_NoShortCuts

Auch wenn Unterstreichungen in den Intuitexten enthalten sind, werden diese generell ingnoriert. Es werden keine Unterstriche gezeichnet und keine Tastaturkürzel berücksichtigt.

RIA_Debug

Wenn der Debug-Modus aktiviert ist, wird bei jedem Fehler, der in der Ressource-Bibliothek auftritt, eine entsprechende Fehlermeldung auf dem Bildschirm angezeigt. Die Bedeutungen der Fehlermeldungen finden Sie im Anhang dieser Referenz.

Hinweis: Da sich die Organisation des Speicherbereiches ändern kann, der durch die Adresse in ApplAddr lokalisiert wird, sollte nie versucht werden den Inhalt zu identifizieren und direkt darauf zuzugreifen, da ständig Neuerungen und Änderungen programmiert werden.

Beispielprogramm:

```
#include <stdio.h>
#include <stdlib.h>
#include <clib/exec_protos.h>
#include <clib/intition_protos.h>
#include <clib/graphics_protos.h>
#include <pragmas/exec_pragmas.h>
#include <pragmas/intuition_pragmas.h>
#include <pragmas/graphics_pragmas.h>
#include <rct/rctdef.h>
#include <rct/rctpragmas.h>
#include <rct/rcttagfuncs.h>

struct IntuitionBase *IntuitionBase;
extern struct Library *Sysbase;

struct EasyStruct rcterror = {
  sizeof( struct EasyStruct), 0, "RCT-Demo",
  "%s", "OK"};

APTR RctBase, ApplAddr;
```

```
void main ( void)
{

if( IntuitionBase = OpenLibrary( ( UBYTE *)"intuition.library",0L))
  {
  if( RctBase = OpenLibrary( "rct.library", 16L))
    {
    if( ApplAddr = R_InitApplTags( TAG_DONE))
      {
      ... /* Hier Hauptprogramm einfügen */
      R_ExitAppl( applicationaddr);
      }
    else
      {
      EasyRequest( NULL, &rcterror, NULL,"Kann RCT-Library nicht initialiseren");
      CloseLibrary( ( struct Library *)rctbase);
      CloseLibrary( ( struct Library *)IntuitionBase);
      exit( -1);
      }
    }
  else
    {
    EasyRequest( NULL, &rcterror, NULL, "Kann RCT-Library nicht öffnen");
    CloseLibrary( ( struct Library *)IntuitionBase);
    exit( -1);
    }
  }
CloseLibrary( ( struct Library *)rctbase);
CloseLibrary( ( struct Library *)IntuitionBase);
exit ( 0);
}
```

R_ExitAppl (V16)

error = R_ExitAppl(appladdr);
D0 A0

WORD error;
APTR appladdr; /* die von R_InitAppl gelieferte Adresse */

Gibt alle, mit den Bibliotheks-Funktionen angeforderten Ressourcen wieder frei. Geöffnete Screens, Fenster, Dialoge und Menüleisten werden automatisch geschlossen. Eventuell eingerichtete IDCMPorts für Fenster werden geschlossen und freigegeben. Geladenen RCT-Dateien werden aus dem Speicher entfernt.

Beispiel: (siehe R_InitAppl).

Die Grundlagen allen Schaffens - Screens und Fenster

Für jede Text und Grafikausgabe muß ein Ausgabemedium verfügbar sein. Auf dem Amiga sind es Screens und hauptsächlich Fenster. Screens werden dazu benutzt, um verschiedene Pixel-Auflösungen, parallel zu anderen Programmen unterschiedliche Darstellungsmodi und unterschiedliche Farbbildschirme zu ermöglichen. Fenster sind immer ein Ausschnitt (Bildbereich) aus dem tatsächlich vorhandenen, und nicht aus dem nur sichtbaren Screenbereich. Da unter AMIGA OS 2.0 Screens größer als der sichtbare Monitorbereich sein können, ist es auch möglich, Fenster größer als den sichtbaren Monitorbereich zu machen. Sie können allerdings nie größer werden, als es der Screen ist, auf dem sie dargestellt werden.

R_ShowScreen (V16)

R_ShowScreenTags (V16)

```
screen = R_ShowScreen( appladdr, taglist);
  D0                    A0        A1

screen = R_ShowScreenTags( appladdr, tag, ...);
```

```
struct Screen    *screen;   /* Rückgabewert */
APTR             appladdr;  /* Rückgaberwert der Funktion R_InitAppl */
strukt TagItem   *taglist;  /* Liste der Einstellparameter */
```

Die Funktion versucht einen Screen zu öffnen und liefert die Adresse auf die „Screen-Struktur" zurück. Ganz ohne Parameterangaben wird automatisch eine Kopie des Workbench-Screens angefertigt. Sobald Parameter angegeben werden, ohne daß man alle nötigen Paramterangaben macht, muß der Tag-Identifier „RSA_LikeWorkbench", mitangegeben werden.
Die Funktion ermittelt automatisch die Darstellungsmodi des Workbench-Screens (PAL oder NTSC) und überträgt diese auf den zu öffnenden Screen. Dadurch werden die unschönen Effekte vermieden, die auftreten, wenn zum Beispiel ein Pal-Screen ungewollt auf einem NTSC-Display geöffnet wird. Teilweis sind dann sogar Bildbereiche des Pal-Screens verdeckt und man wäre normalerweise gezwungen, die Standardauflösung mit den Screen-Mode-Preferences zu verändern.

Bei Erfolg erhält man die Adresse des geöffneten Screens (struct Screen *). Diese benötigt man, um den Screen wieder zu schließen.

Der Funktion **R_ShowScreenTags** werden die einzelnen Parameter direkt übergeben. Man muß keine TagItem-Liste im Speicher aufbauen!

Es handelt sich hierbei um keine zusätzliche Library-Funktion, sondern sie ist in der Header-Datei „RCTTAGFUNCS.H" als C-Funktion mit variabler Parameterliste deklariert und ruft ihrerseits die Funktion **R_ShowScreen** auf.

> **Hinweis:** Die Funktion **R_ShowScreenTags** ist nicht für alle Programmiersprachen nutzbar.

Im Folgenden werden die einzelnen Einstell- und Übergabeparameter erläutert.

RSA_LikeWorkbench
Fertigt eine exakte Kopie der Workbench an. Bei übergabe weiterer Parameter überschreiben diese die Parameter des „Clones". Es ist somit möglich, nur Teile zum Beispiel die Höhe und Weite des Workbench-Screens zu übernehmen.

RSA_Left
Bestimmt ebenfalls die Position der linken, oberen Ecke des Screens. Dieser Parameter hat erst unter Amiga OS 2.0 eine Wirkung. Bei älteren Betriebssystemversionen wird er einfach ignoriert.

RSA_Top (auch unter Kick 1.3)
Bestimmt die Position der linken oberen Ecke des Screens. Linke obere Ecke des Screens ist immer relativ zum Bildfeld von Intuition und wird immer in der Pixelgröße der entsprechend gewählten Auflösung gesetzt.

RSA_Width (auch unter Kick 1.3)
Die Breite des Screen-RastPorts in Pixeln.

RSA_Height (auch unter Kick 1.3)
Die Höhe des Screen-RastPorts in Pixeln. Mit den Konstanten STDSCREENWIDTH und STDSCREENHEIGHT werden die Standardwerte für Weite und Höhe eingesetzt. Diese Werte richten sich nach den IntuitionBase-Member „NormalDisplayRows" und „NormalDisplayColumns". Sie beziehen sich immer auf die für den Text-Overscan eingestellte Größe.

RSA_Depth (auch unter Kick 1.3)
Die Anzahl der Bitplanes für den Screen. Je mehr Bitplanes angegeben werden, desto mehr Farben können auf dem Screen dargestellt werden. Um übertriebene Angaben zu vermeiden, sollte geprüft werden, wieviele Bitmaps maximal für die entsprechende Auflösung genutzt werden können. Verwenden Sie für die Auswahl einfach das Auflösungsauswahlfenster.

RSA_DetailPen (auch unter Kick 1.3)

Die Farbnummer für die Screen Details. Das ist die Textfarbe im Screentitel und Farbe der Gadgets. Der Standardwert hierfür ist 0.

RSA_BlockPen (auch unter Kick 1.3)

Die Farbnummer für die Block-Füllungen der Screentitelleiste. Der Standardwert hierfür ist 1.

RSA_Type (auch unter Kick 1.3)

Die Konstante für den Screentyp. Mögliche Werte sind:

CUSTOMSCREEN - das ist der Wert für einen eigenen Screen. Es handelt sich dann nicht um einen System-Screen.

CUSTOMBITMAP - der Screen bekommt eine eigene Bitmap in RSA_BitMap zugewiesen. Das System allociert dann keine eigene Bitmap für den Screen.

SCREENBEHIND - der Screen wird im Hinergrund (hinter allen anderen Screens) geöffnet. Man hat dann die Möglichkeit, die Farben korrekt einzustellen oder kann eine Grafik laden und den Screen mit der Funktion "ScreenToFront„ in den Vordergrund bringen, wenn der Aufbau komplett abgeschlossen ist.

SCREENQUIT - für den Screen werden keine Systemgadgets und keine Titelzeile angezeigt.Wenn für die Fenster, die auf dem Screen dargestellt werden sollen, das Flag WFLG_RMBTRAP eingestellt ist, wird auch während des Arbeitens mit dem Screen keine Titelleiste gezeichnet. Ist das WFLG_RMBTRAP-Flag für Fenster nicht gesetzt oder werden mit dem IDCMP_MENUVERIFY, Menüoperationen nicht abgefangen, kann mit SCREENQUIT verhindert werden, daß die Screen-Titelleiste gelöscht wird. Unter Amiga OS 1.3 wird die Titelleiste beim Öffnen des Screen allerdings immer gelöscht. Ab Amiga OS 2.0 passiert dies nicht mehr.

RSA_Font (auch unter Kick 1.3)

„ti_Data" muß die Adresse einer „TextAttr-Struktur" übergeben werden. Der Dort spezifizierte Font wird im Screentitel, in den Menüs und in den Fenstern, die auf dem Screen zur Anzeige gebracht werden, eingesetzt.

Hinweis: Die Funktion öffnet den Font nicht automatisch. Ist der gewünschte Font nicht im System vorhanden, muß er zuvor mit der Funktion OpenDiskFont (Diskfont-Bibliothek) geladen werden. Der Font darf dann solange nicht wieder freigegeben werden, wie der Screen angezeigt bleibt.

RSA_Title (auch unter Kick 1.3)

Der Text, der in der Screen-Titelleiste angezeigt werden soll. Ein normaler C-String mit einem Nullbyte am Ende oder einfach nur Null, wenn kein Text angezeigt werden soll.

RSA_BitMap (auch unter Kick 1.3)

Wenn in RSA_Type CUSTOMBITMAP deklariert wurde, muß in RSA_BitMap die Adresse der entsprechenden BitMap-Struktur überben werden.

RSA_DisplayID (auch unter Kick 1.3)

Hier wird die Screenauflösung angegeben. Die genauen Definitionen sind in der Include-Datei „graphics/displayinfo.h" beschrieben.
Beispiel:
HIRES
HIRES|LACE
...
Bei Amiga OS-Versionen >= 2.0 werden automatisch die für die angegebene Auflösung eingestellten TextOverscanwerte in Weite und Höhe der NewScreen-Struktur übernommen. Sind in der angegebenen Parameterliste (egal an welcher Stelle der Liste) Angaben für Weite und Höhe des Screens enthalten, haben diese jedoch Priorität.
Für Amiga OS-Versionen < 2.0 werden Normalwerte ohne Berücksichtigung einer Overscaneinstellung aus einer Tabelle übernommen. Dabei wird zwischen PAL- und NTSC-Auflösungen unterschieden.

RSA_DisplayIDStruct (auch unter Kick 1.3)

In „ti_Data" wird die Adresse einer „R_ScrModeReq-Struktur" übergeben. Man kann so den Anwender des Programmes mit Hilfe des Auflösungsdialolges eine Auswahl treffen lassen und die zurückgelieferten Werte direkt übergeben.

Dieser Parameter kann nicht in der normalen Intuition-Bibliothek verwendet werden.

RSA_ErrorCode (auch unter Kick 1.3)

In „ti_Data" wird die Adresse eines ULONGS erwartet, in der ein entsprechender Error-Code untergebracht werden kann, wenn die Funktion nicht korrekt ausgeführt wird. Falls die Funktion Erfolg hat, wird Null übergeben andernfalls:
OSERR_NOMONITOR - der gewählte Display-Modus ist auf dem angeschlossenen Monitor nicht darstellbar.
OSERR_NOCHIPS - es werden neuere Custom-Chips (ECS, AA...) für den gewünschten Display-Modus benötigt.
OSERR_NOMEM - nicht genügend Fast-Ram verfügbar.
OSERR_NOCHIPMEM - nicht genügend Chip-Mem verfügbar.

OSERR_PUBNOTUNIQUE - der eingesetzte Name für den Public-Screen ist bereits vergeben.
OSERR_UNKNOWNMODE- der gewünschte Display-Modus ist auf dem System nicht verfügbar.

Die nachfolgenden Parameter werden nur unter Amiga OS 2.0 und höher berücksichtigt. Die Verwendung unter Amiga OS 1.3 wird ignoriert.

RSA_Colors
In „ti_Data" wird die Adresse einer „ColorSpec-Struktur" übergeben (siehe intuition/intuition.h). Mit dieser Strukturliste werden die Farbregister für den Screen schon vor dem Anzeigen initialisiert. Als Kennzeichnung des letzten Listeneintrages muß im Member „ColorIndex" der Wert -1 übergeben werden.

```
struct ColorSpec {
    WORD ColorIndex;
    UWORD      Red;    // unter Intuition < V 37 werden nur 6 Bit beachtet
    UWORD      Green;
    UWORD      Blue;
    };
```

RSA_SysFont
In „ti_Data" wird eine der beiden, mit den System-Preferences vorgenommenen, Einstellungen ausgewählt. Die Angabe dieses Parameters hat Priorität vor den Angaben in „RSA_Font".
Mögliche Werte:
0 - der Normale Standardfont mit fester Weite (nicht proportional).
1 - der in der Workbench genutzte Font (kann proportional sein).

RSA_PubSig
„ti_Data" enthält eine Signalnummer, über die eine Nachricht geschickt werden kann, wenn das letzte Fremd-Fenster auf dem Screen geschlossen wird. Mit Fremd-Fenster ist ein Fenster eines anderen Programmes (beispielsweise eines kontextsensitiven Hilfesystems) gemeint.

RSA_PubTask
Die Adresse des Task, der die Nachricht empfangen soll.

RSA_PubName
Ist in „ti_Data" ein Screen-Name angegeben, wird der Screen als Public-Screen mit Status „PRIVAT" geöffnet. Zum Ändern des Status muß die Intuition-Funktion PubScreenStatus benutzt werden.

RSA_DClip

„ti_Data" enthält die Adresse einer Rechteck-Struktur, die die Ausmaße für das Display-Clipping-Rechteck angibt. Nährere Angaben finden Sie bei der Erklärung zur Intuition-Funktion „QueryOverscan".

RSA_Overscan

Die Konstante in „ti_Data" spezifiziert den Overscan-Modus für den Screen. Dieser Parameter macht es sehr einfach einen Screen im Overscan-Modus zu öffnen. Der Overscan für die entsprechende Auflösung wird mit den „System-Preferences" eingestellt.
Mögliche Werte sind:
OSCAN_TEXT - Diese Einstellung wird auch vom Workbench-Screen verwendet. Mit dieser normalen Einstellung bleibt noch alles sichbar.

OSCAN_STANDARD - Die Ecken dieses Rechteckes liegen so gerade außerhalb des sichbaren Monitorbildes. Dieser Modus ist sehr gut für Zeichenprogramm und Spiele geeignet.

OSCAN_MAX - Dies ist das maximal Mögliche, was von den Funktionen der Graphics-Bibliothek noch einigermaßen gut verwaltet werden kann.

OSCAN_VIDEO - In diesem Modus läßt sich nicht mehr gut arbeiten. Er ist für Video-Anwendungen geeignet.

RSA_ShowTitle

Der eingesetzte Screentitel soll angezeigt werden oder nicht. „ti_Data" ist ein Boolean.

RSA_Behind

Dieser Parameter ist equivalent zu **SCREENBEHIND** in „RSA_Type". „ti_Data" ist ein Boolean.

RSA_Quiet

Dieser Parameter ist equivalent zu **SCREENQUIET** in „RSA_Type". „ti_Data" ist ein Boolean.

RSA_AutoScroll

Ist „ti_Data" TRUE soll der Screen automatisch gescrollt werden, wenn mit der Maus die Kanten des Screens berührt werden. Die Einstellung setzt einen Screen voraus, der größer ist als die sichtbare Monitordarstellung. Falls dies nicht der Fall ist, wir der Parameter ignoriert.

RSA_Pens

In „ti_Data" wird die Adresse eine Feldes vom Type UWORD übergeben. Die Werte legen die Farbspezifikationen für die „DrawInfo-Struktur" des Screen fest. Mit Hilfe der Angaben in dieser Struktur wird der 3D-Effekt in den System-Gadgets erzeugt.
Mit Übergaben eines minimalen Array, das lediglich den Wert „ ~0 " enthält, kann der Standard-3D-Look erzeugt werden.
Mit R_ShowScreen wird ab Amiga OS 2.0 automatisch ein 3-D Effekt erzeugt.

RSA_FullPalette

„ti_Data" ist ein Boolean. Unter Amiga OS 1.3 stehen 7 Farbwerte für einen Screen zur verfügung. Farbnummern 0-3 sind die normalen Screen Farben und Farbnummern 17-19 sind die Farben für Sprites. Wird ein Screen geöffnet, initialisiert zuerst die Graphics-Bibliothek die Farbwerte, bevor sie durch die Werte, die in den Preferences eingestellt wurden, überschrieben werden. Ab Amiga OS 2.0 können alle 32 Farben überschrieben werden.

R_RemScreen

error = R_RemScreen(appladdr, screen);
D0 A0 A1

```
WORD          error;     /* Rückgabewert */
APTR          appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Screen *screen;   /* Adresse des Screens der geschlossen werden soll */
```

Schließt den angegebenen Screen. Bevor der Screen vom Bildschirm verschwindet werden alle Windows und Requester auf ihm geschlossen, so daß keine „Fensterleichen" entstehen.

Achtung:
Ist auf dem Screen ein „Public-Window" geöffent, darf der Screen nicht geschlossen werden. Im „Debug-Modus"wird die Fehlermeldung **RREMW_WINUNKNOWN** angezeigt (siehe R_IntiAppl). Als Rückgabewert erhält man dann den gleichen Fehler. Im Erfolgsfall wird 0 zurückgeliefert. Genaue Angaben zu den Fehlermeldungen finden Sie im Anhang.

Ein Beispiel für die Screenfunktionen:

```
#include <exec/types.h>
#include <intuition/intuition.h>
#include <libraries/dosextens.h>
#include <functions.h>
#include <stdio.h>
#include „rctdef.h“struct IntuitionBase *IntuitionBase;

APTR ApplBase;
APTR RctBase;

void main( void)
{
struct Window *win;
SHORT handle,quit = 1;
struct IntuiMessage msg;
char *x;

if( ! (RctBase = (APTR)OpenLibrary( ( UBYTE *)“rct.library“,16L)))
    {
    printf(„Can‘t open libraries\n“);
    return;
    }
if( ApplBase = R_InitApplTags( TAG_DONE))
    {
    R_ShowScreenTags( ApplBase, RSA_LikeWorkbench, 1, TAG_DONE);

    /* Hier Hauptprogramm einfügen! */

    R_ExitAppl( ApplBase);
    }
CloseLibrary( ( struct Library *)RctBase);
retrun 0;
}
```

R_ShowWindow (V16)

R_ShowWindowTags (V16)

window = R_ShowWindow(appladdr, taglist);
D0 A0 A1

window = R_ShowWindowTags(appladdr, tag, ...);

```
struct Window  *window; /* Rückgabewert */
APTR           appladdr; /* Rückgabewert der Funktion R_InitAppl */
strukt TagItem *taglist; /* Liste der Einstellparameter */
```

Mit Hilfe der Funktion R_ShowWindow wird ein Fenster geöffnet. Größe, Attribute und Flags des Fensters sind von den Default-Einstellungen abhängig. Ohne Änderung der Default-Einstellungen, wird, direkt nach Aufruf von *R_InitAppl*, das Fenster in voller Workbenchgröße geöffnet.
Ohne Angabe von Parametern öffnet sich ein Fenster, dessen Attribute und Größe von den zuvor durchgeführten Fenster-Funktionen bestimmt werden.

Die Funktion liefert die Adresse des Fensters (struct Window *) oder im Fehlerfall 0.

Der Funktion *R_ShowWindowTags* werden die einzelnen Parameter direkt übergeben. Es handelt sich hierbei um keine zusätzliche Library-Funktion, sondern sie ist in der Header-Datei „RCTTAGFUNCS.H" als C-Funktion mit variabler Parameterliste deklariert und ruft ihrerseits die Funktion R_ShowWindow auf.

> **Hinweis:** Die Funktion **R_ShowWindowTags** ist nicht für alle Programmiersprachen nutzbar.

Im Folgenden werden die einzelnen Einstell- und Übergabeparameter erläutert.

RWA_SCRCPY (auch unter Kick 1.3)
Kopiert die Weiten und Höhen des Default-Screens in die des Default-Windows. Wenn keine weiteren Parameter folgen wird das Fenster in der Größe des Default-Screens geöffnet.

RWA_Left (auch unter Kick 1.3)
Die X-Koordinate für die linke, obere Ecke des Fensters.

RWA_Top (auch unter Kick 1.3)
Die Y-Koordinate für die linke, obere Ecke des Fensters.

RWA_Width (auch unter Kick 1.3)
Die Breite des Fensters, relativ zur durch RWA_Left bestimmten X-Startkoordinate.

RWA_Height (auch unter Kick 1.3)
Die Höhe des Fensters, relativ zur durch RWA_Top bestimmten Y-Startkoordinate.

RWA_DetailPen (auch unter Kick 1.3)
Farbnummer, in der Gadgets und der Text in der Titelzeile gezeichnet werden sollen. Bei Angabe von -1 wird die Einstellung aus dem Screen übernommen.

RWA_BlockPen (auch unter Kick 1.3)
Farbnummer, in der die Titelleiste und die Umrandung des Fensters gezeichnet werden soll. Bei Angabe von -1 wird die Einstellung aus dem Screen übernommen.

RWA_IDCMP (auch unter Kick 1.3)
Die Nachrichtenübermittlung von Intuition funktoniert über den „Intuition Direkt Communication Message Port". Für jedes geöffnete Fenster kann ein eigener Port eingerichtet werden, für den dann unterschiedliche IDCMP-Flags eingestellt werden können. Das bedeutet, daß nicht immer alle tatsächlich auftretenden Nachrichten auch ausgewertet werden müssen, sondern man kann sich für eine gewisse Zusammenstellung entscheiden. Auch nach dem Öffnen des Fensters läßt sich die Zusammenstellung der Flags ändern. Hierzu dient die Funktion „ModifyIDCMP". Die Flags sind in der Header-Datei „intuition/intuition.h" definiert. Mit dem „|"-Zeichen lassen sich die Flags in der Liste kombinieren. Beispiel

```
(ICDMP_GADGETDOWN|IDCMP_GADGETUP)
```

Mit der Bibliotheksfunktion R_GetMsg werden die eintreffenden Nachrichten empfangen. Geliefert bekommt man die Adresse einer Struktur, die wie folgt aufgebaut ist:

```
struct R_IntuiMessage
{
  struct IntuiMessage *IMsg; // die Origianl Nachricht von Intuition.
  ULONG  R_Class;            // die gefilterte Nachrichten-Klasse
  ULONG  R_Subclass;         // der gefilterte Nachrichten-Unterklasse
  ULONG  R_Code;             // Der gefiltere Nachrichten-Code
  ULONG  Reserve1;
  ULONG  Reserve2;
}
```

Es existieren folgende Nachrichtenklassen:

IDCMP_SIZEVERIFY
: Sobald das Size-Gadget im Fensterrahmen gedrückt wird, erhält man diese Nachricht

IDCMP_NEWSIZE
: Wenn das Fenster vergrößert oder verkleinert wurde.

IDCMP_REFRESHWINDOW
: Der Fensterinhalt muß erneuert werden.

IDCMP_MOUSEBUTTONS
: Die Maustasten wurden gedrückt.

IDCMP_MOUSEMOVE
: Die Maus hat sich bewegt. Die Mauskoordinaten relativ zur oberen, linken Ecke des Fensters werden im Struktur-Member MouseX und MouseY abgelegt.

IDCMP_GADGETDOWN
: Über einem Gadget wurde die linke Maustaste gedrückt.

IDCMP_GADGETUP
: Die Maustaste wurde wieder über einem Gadget losgelassen. Zuvor muß also die Maustaste über einem Gadget gedrückt worden sein.

IDCMP_REQSET
: Innerhalb des Fenster, wurde ein Requester geöffnet.

IDCMP_MENUPICK
: Es fand eine Menüauswahl statt.

IDCMP_CLOSEWINDOW
: Das Close-Gadget in der linken oberen Ecke des Fensters wurde angewählt.

IDCMP_RAWKEY
: Eine Taste auf der Tastatur wurde gedrückt oder losgelassen. Im Member „R_Subclass" werden der Scancode der Taste, und der ASCII-Code, falls möglich, geliefert.

IDCMP_REQVERIFY
: Die Nachricht trifft ein, bevor ein Requester gezeichnet wird.

IDCMP_REQCLEAR
: Ein Requester wurde geschlossen.

IDCMP_MENUVERIFY
: Der Anwender ist im Begriff, eine Menüauswahl zu treffen. Man hat dann Gelegenheit, alle Grafikausgaben für das Fenster abzuschließen, bevor das Menü angezeigt wird.

IDCMP_NEWPREFS
: Die Preferences-Einstellungen wurden verändert.

IDCMP_DISKINSERTED
: Eine Diskette wurde in ein Laufwerk eingelegt.

IDCMP_DISKREMOVED
: Eine Diskette wurde aus einem Laufwerk entfernt.

IDCMP_ACTIVEWINDOW
Das Fenster wurde aktiviert.
IDCMP_INACTIVEWINDOW
Das Fenster wurde inaktiviert.
IDCMP_DELTAMOVE
Die Maus wurde bewegt. Im Member `MouseX` und `MouseY` werden die Koordinaten relativ zur letzten Mausposition übermittelt.
IDCMP_VANILLAKEY
Eine Taste wurde gedrückt. Im Memer R_Subclass wird der ASCII-Code der gedrückten Taste übermittelt. Leider wird kein SCAN-Code der Taste übermittelt, so daß man keine Nachricht beim Drücken einer Sondertaste (z.B. Funktionstaste...) erhält. Es ist besser mit IDCMP_RAWKEY die Tastaturnachrichten abzufragen.
IDCMP_INTUITICKS
Wenn möglich, erhält man 10 Mal pro Sekunde eine IDCMP_INTUITICKS-Nachricht. Dauert die Auswertung der Nachricht länger als eine 1/10 Sekunde, stauen sich die Nachrichten nicht an.

RWA_Flags (auch unter Kick 1.3)
Mit den nachfolgend beschriebenen Flags wird das Aussehen des Fensters definiert.

WFLG_SIZEGADGET
Das Fenster soll vergrößerbar sein.
WFLG_DRAGBAR
Das Fenster wird verschiebbar.
WFLG_DEPTHGADGET
Das Fenster kann mit der Maus in den Vorder- oder Hintergund geklickt werden.
WFLG_CLOSEGADGET
Ein Schließfeld wird in der Titelzeile des Fensters angezeigt.
WFLG_SIZEBRIGHT
Der Platz für das Size-Gadget wird von der rechten Seite des Fensters reserviert.
WFLG_SIZEBBOTTOM
Der Platz für das Size-Gadget wird von der unteren Kante des Fensters reserviert.
WFLG_SMART_REFRESH
WFLG_SIMPLE_REFRESH
WFLG_SUPER_BITMAP
Die drei Flags bestimmen den Type des Layers, der für das Fenster benutzt werden soll.
WFLG_BACKDROP
Das Fenster steht immer im Hintergrund auf dem Screen. Es dient nur zur Nachrichtenmitteilung. Normalerweise kombiniert man dieses Flag mit der Eigenschaft WFLG_BORDERLESS.

WFLG_REPORTMOUSE
Dieser Flag muß eingeschaltet sein, wenn mit der Intuition-Funktion „Report-Mouse" oder mittels IDCMP_MOUSEMOVE, Mausbewegungsnachrichten geliefert werden sollen.

WFLG_GIMMEZEROZERO
Bei gesetztem WFLG_GIMMEZEROZERO-Flag werden die Ränder des Fensters in einem von Intuition zusätzlich angelegten Layer verwaltet. Man muß dann nicht mehr auf ein Überschreiben (Zerstören) der Ränder achten. Allerdings werden solche Fenster wesentlich langsamer aufgebaut und neugezeichnet. Der Maus-Nullpunkt liegt nicht auf der linken, oberen Ecke des Fensters, sondern in der linken oberen, Ecke der Zeichenfläche.

WFLG_BORDERLESS
Das Fenster soll ohne Randkomponenten gezeichnet werden.

WFLG_ACTIVATE
Das Fenster wird beim Öffnen gleich zum aktiven Fenster auf dem Screen.

WFLG_RMBTRAP
Ist WFLG_RMBTRAP gesetzt, werden IDCMP_MOUSEBUTTON-Nachrichten beim Drücken der rechten Maustaste geliefert. Eine Menüauswahl wird unterdrückt.

WFLG_NOCAREREFRESH
Im Programm werden die beiden Intuition-Funktionen „BeginRefresh" und „EndRefresh" nicht aufgerufen, wenn bestimmte Bereiche des Fensters neu aufgebaut werden sollen.

RWA_Gadgets (auch unter Kick 1.3)
„ti_Data" zeigt auf die Adresse einer Gadget-Liste, die im Fenster angezeigt werden sollen.

RWA_Checkmark (auch unter Kick 1.3)
In „ti_Data" muß die Adresse einer eigenen „Image-Struktur" (struct Image *) übergeben werden, die das „Checkmark-Image" (abgehakt) in Menüleisten enthält.

RWA_Title (auch unter Kick 1.3)
Der Text, der in der Titelzeile des Fensters angezeigt werden soll.

RWA_CustomScreen (auch unter Kick 1.3)
Das Fenster soll auf einem bestimmten Screen geöffnet werden. Man muß dazu noch die Adresse eines geöffneten Screens angeben. Diese wird von den Funktionen **R_ShowScreen** zurückgeliefert.

RWA_SuperBitMap (auch unter Kick 1.3)
Wenn das Flag „WFLG_SUPER_BITMAP" gesetzt ist, muß in „ti_Data" die Adresse auf die „BitMap-Struktur" (struct BitMap *) enthalten sein.

RWA_MinWidth (auch unter Kick 1.3)
Die minimale Breite des Fensters.

RWA_MinHeight (auch unter Kick 1.3)
Die minimale Höhe des Fensters.

RWA_MaxWidth (auch unter Kick 1.3)
Die maximale Breite des Fensters.

RWA_MaxHeight (auch unter Kick 1.3)
Die maximale Höhe des Fensters.

Für die nachfolgend erklärten Parameter wird in „ti_Data" ein Boolean erwartet.

RWA_AutoAdjust (auch unter Kick 1.3)
Das Fenster kann automatisch in der Größe und in der Position geändert werden, wenn es durch die angegebenen Größen nicht auf den Screen paßt.
Es werden allerdings die Einstellungen für „RWA_MinWidth" und „RWA_MinHeight" berücksichtigt.

RWA_SizeGadget (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_SIZEGADGET**.

RWA_DragBar (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_DRAGBAR.**

RWA_DepthGadget (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_DEPTHGADGET.**

RWA_CloseGadget (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_CLOSEGADGET.**

RWA_BackDrop (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_BACKDROP.**

RWA_ReportMouse (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_REPORTMOUSE.**

RWA_NoCareRefresh (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_NOCAREREFRESH.**

RWA_Borderless (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_BORDERLESS.**

RWA_Activate (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_ACTIVATE.**

RWA_RMBTrap (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_RMBTRAP**.

RWA_SimpleRefresh (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_SIMPLE_REFRESH.**

RWA_SmartRefresh (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_SMART_REFRESH.**

RWA_SizeBRight (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_SIZEBRIGHT.**

RWA_SizeBBottom (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_SIZEBBOTTOM.**

RWA_GimmeZeroZero (auch unter Kick 1.3)
Äquivalent zum Flag **WFLG_GIMMEZEROZERO.**

Die folgenden Einstellungen haben nur unter Amiga OS 2.0 eine Auswirkung:

RWA_ScreenTitle
Der in „ti_Data" angegebene Texte (String-Adresse) wird im Screentitel angezeigt, wenn das Fenster aktiv ist.

RWA_InnerWidth

RWA_InnerHeight
Mit diesen Angaben kann das tatsächlich vorhandene Zeichenfeld unabhängig von den Größen der Fensterränder eingestellt werden. Man sollte das Flag RWA_AutoAdjust angegeben haben, damit die Größe des Fensters automatisch verändert werden kann.

RWA_PubScreenName
In „ti_Data" wird der Name (String-Adresse) des Public-Screens angegeben, auf dem das Fenster geöffnet werden soll.

RWA_PubScreen
Das Fenster soll auf dem Public-Screen geöffnet werden, der durch die Screen-Adresse in „ti_Data" spezifiziert wird. Damit Sie die Garantie haben, daß der Screen lange genug geöffnet bleibt, müssen Sie:
- den Screen selbst geöffnet haben
- ein anderes Fenster auf dem Screen geöffnet haben
- mit der Intuition-Funktion LockPubScreen das ungewollte Schließen des Screens verhindern.

Wird in „ti_Data" Null übergeben, öffnet sich das Fenster auf dem Default-Public-Screen.

RWA_PubScreenFallBack
Das Fenster wird auf dem Default-Public-Screen geöffnet, wenn der gewünschte Public-Screen nicht vorhanden ist. „ti_Data" ist ein Boolean.

RWA_Zoom
In „ti_Data" wird die Adresse eines „WORD-Feldes" erwartet. Die Werte intialisieren die alternative Größe des Fensters, die eingestellt wird, wenn man das Zoom-Gadget anklickt. Auch wenn das Flag „RWA_SizeGadget" nicht gesetzt ist, wird dadurch das Zoom-Gadget angezeigt.

RWA_BackFill
In „ti_Data" wird die Adresse einer „Hook-Struktur" erwartet, die von den Funktionen der Layers-Bibliothek aufgerufen wird. Man hat dadurch die Möglichkeit, den Fensteruntergrund mit einer Grafik zu versehen. Die Verwaltung übernimmt die Layers-Bibliothek.

R_RemWindow (V16)

error = R_RemWindow(appladdr, window)

```
SHORT        error;     /* Rückgabewert */
APTR         appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Window *window;  /* Adresse des zu schließenden Fensters */
```

Die Funktion löscht ein mit *R_ShowWindow* gezeichnetes Fenster vom Bildschirm. Dabei werden eventuell für dieses Fenster gesetzte Menüleisten und innerhalb des Fensters angezeigte Requester automatisch gelöscht.

Die Funktion wird beim Aufruf von *R_ExitAppl* zum Beenden des Programms ebenfalls für jedes, noch geöffnete Fenster aufgerufen!

R_Raw2Asc (V16)

ascii = R_Raw2Asc(appladdr, message);
D0 A0 A1

```
UBYTE         ascii;     /* Der ermittelte ASCII-Wert */
APTR          appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Message *message; /* Adresse einer IntuiMessage-Struktur */
```

Konvertiert einen RawKey-Code (Tastatur-Scancode) in einen ASCII-Code. Internwird dazu die Funktion RawKeyKonvert benutzt, damit die Konvertierung auch Länderspezifisch vorgenommen wird. Desweiteren werden Dead-Keys berücksichtigt.

R_SetSysRequest (V16)

error = R_SetSysRequest(appladdr, window);
D0 A0 A1

```
SHORT         error;     /* Rückgabewert */
APTR          appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Window *window;   /* Adresse eines angezeigten Fensters */
```

Die Funktion setzt das Ausgabefenster für DOS-Fehlermeldungen auf das in *window* angegebene und bereits angezeigte Fenster. Wird diese Funktion nicht aufgerufen, werden alle DOS-Fehlermeldungen auf dem Workbench-Screen ausgegeben.

> **Hinweis:**Beim Beenden des Programmes muß die Veränderung, die in der Prozess-Struktur vorgenommen wurde, wieder rückgängig gemacht werden. Dies geschieht automatisch beim Aufruf der Funktion *R_ExitAppl* zum Freigeben der angeforderten Ressourcen.

R_ResetSysRequest (V16)

error = ResetSysRequest(appladdr)
D0 A0

```
SHORT  error;      /* Rückgabewert */
APTR   appladdr;   /* Rückgabewert der Funktion initappl */
```

Diese Funktion setzt das mit *R_SetSysRequest* gesetzte Ausgabefenster zurück. Sie wird mit Aufruf von exitappl automatisch ausgeführt. Muß demnach nicht unbedingt explizit aufgerufen werden!

R_Create (V16)

R_CreateTags (V16)

address = R_Create(appladdr, type, taglist);
D0 A0 D1 A1

adresse = R_Createtags(appladdr, type, tag1, ...);

```
APTR          adresse;    /* Rückgabewert je nach angefordertem Objecttyp */
APTR          appladdr;   /* Der von der Funkiotn R_InitAppl zurückgelieferte Wert */
LONG          type;       /* Der betreffende Objectype (siehe unten) */
struct TagItem *taglist;  /* Tag-Parameterliste */
```

Die Funktion dient zum Anlegen und Kopieren von einzelnen Ressource-Teilen.

Der Parameter **type** kann folgende Werte annehmen:

R_Border

R_Text

R_Gadget

R_MenuItem

R_Menu

R_Request
Die Angaben in der Tag-Parameterliste beziehen sich auf eine Requester-Struktur.

Folgende Tag-Parameter lassen sich angeben:

R_Dup_Res
Das in Typ angegebene Object soll dupliziert werden.

RNo_TxtAttr
TextAttribut-Strukturen nicht berücksichtigen.
Beim duplizieren werden die Adressen der Quell-Objecte eingesetzt.

RNo_IntTxt
IntuiText-Strukturen nicht berücksichtigen.
Beim duplizieren werden die Adressen der Quell-Objekte eingesetzt.

RNo_IntTxtStr
Die Texte von IntuiText-Strukturen sollen nicht berücksichtigt werden. Beim duplizieren werden die Adressen der Quell-Objecte eingesetzt. Ist dieses Flag beim duplizieren gesetzt, wird der Text bei einer eventuellen Änderung auch im Quell-Objekt geändert, da der Text-String nur ein einziges mal im Speicher vorhanden ist.

RNo_Brd
Border-Strukturen sollen nicht berücksichtigt werden.

RNo_BrdData
Die Border-Koordinaten sollen nicht berücksichtigt werden.

RNo_Img
Image-Strukturen sollen nicht berücksichtigt werden.

RNo_ImgData
Die Grafik-Daten der Images sollen nicht berücksichtigt werden. Man spart dadurch Chip-Speicher.

Der Funktion *R_CreateTags* werden die einzelnen Parameter direkt an die Funktion übergeben. Man muß keine TagItem-Liste im Speicher aufbauen! Es ist keine zusätzliche Bibliotheksfunktion sondern in der Schnittstellen-Datei der jeweiligen Programmiersprache als Zwischenschritt vor *R_Create* geschaltet.

R_Destroy (V16)

void R_Destroy(appladdr, object);
A0 A1

```
void            /* Funktion ohne Rückgabewert */
APTR appladdr;  /* Der von der Funkiotn R_InitAppl zurückgelieferte Wert */
APTR object;    /* Adresse eines mit R_Create erzeugten Objectes */
```

Die Funktion gibt das in object spezifizierte, mit der Funktion *R_Create* erzeugte Objekt mit samt seinen Unterobjekten frei. Wird mit *R_Create* beispielsweise ein Requester kopiert, wird auch der Inhalt des Requesters, die Gadgets, Border und Texte mitkopiert, die selbstverständlich auch wieder freigegeben werden müssen. Sie wird mit Aufruf von exitappl automatisch ausgeführt. Muß demnach nicht unbedingt explizit aufgerufen werden!

Die Library-Requester

R_Filer (V16)

handle = R_Filer(appladdr, taglist);
D0 A0 A1

```
WORD          handle;    /* Rückgabewert */
APTR          appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct TagItem *taglist;  /* Lister der Einstellungen */
```

Mit Aufruf dieser Funktion erscheint eine komfortables, vollkommen tastatursteuerbares, reentrantes Dateiauswahlfenster.

Alle Funktionen des Dateiauswahlfensters können mit der Tastatur erreicht werden. Die Gadgets und auch die Texteingabe-Gadgets sind durch Drücken der Ctrl- oder einer der beiden AMIGA-Taste zusammen mit dem jeweiligen Anfangsbuchstaben des Gadgettextes zu erreichen.

Die Liste der Laufwerke und Pfade wird durch Klick auf das Gadget *Disks* oder durch Drücken der rechten Maustaste angezeigt. Ein weiterer Klick auf das Gadget oder Drücken der rechten Maustaste zeigt wieder die Dateien im Fenster an.

Mit den Cursortasten-Auf und -Ab kann der Dateiname oder der Devicename, je nachdem was gerade angezeigt wird, gewechselt werden. Tatsächlich angewählt und in das jeweilige Stringgadget übernommen wird er allerdings erst, wenn Return bestätigt wird. Ist das Texteingabe-Gadget für den Dateinamen aktiviert und wird dann die Return-Taste gedrückt, gilt der Dateiname als ausgewählt und der Requester wird beendet.

Eventuelle Pfadeinstellungen werden beim Wechseln der Devices gespeichert und bei erneutem Wechsel wieder eingestellt.

Der FileRequester ist vergrößerbar und kann so optimal eingestellt werden. Die Textfonts sowohl in den Gadgets als auch im Anzeigefeld werden in der aktuell eingestellten Systemfontart (Screen text) angezeigt. Die Größe darf allerdings 26 Punkt nicht überschreiten. Es soll Leute geben, die dies immer wieder übertreiben. Ist der eingestellte Font so groß, daß 5 Textzeilen auf die gesammte Höhe des Screens nicht mehr angezeigt werden können wird automatisch der Font TOPAZ in der Größe 9 Punkt benutzt.

Die Bumerang Funktion

Der Bumerang bringt eine Vereinfachung bei der Datei-Auswahl von sehr oft benötigten Dateien. Der File-Requester merkt sich hierzu die am häufigsten benötigten Pfade und Dateinamen und bietet diese in einer weiteren Liste zur Auswahl.

Der Aufbau der Bumerang-Sektion innerhalb der RLIB-Configurartions-Datei:

Nach dem Schlüsselwort **R_BUMERANG_STRINGS** folgen die Pfad und Dateiname, die in der Bumerang-Liste angeboten werden sollen.

Die Liste könnte sich nun beliebig verlängern, wenn mit sehr viele unterschiedliche Pfade gearbeitet wird. Damit dies nicht passiert, kann festgelegt werden, wieviele statische und wieviele dynamische Einträge vorhanden sein sollen.

Statische Einträge -
sind immer vorhanden und können über die EXTRA-Auswahl ergänzt und neu arrangiert werden.

Dynamische Einträge -
sind veränderlich. Je nach festgelegter Anzahl dieser Einträge wird beim Hinzufügen der letzte Eintrag gelöscht und ein neuer an den Anfang der Liste aufgenommen.

In der Konfigurationsdatei erkennt man die beiden Bumerang-Typen an dem auf den String folgenden Flag.

STATIC 0
DYNAMIC 1

Folgende Parameter können angegeben werden:

RF_Handle
Der übergebene Handle identifiziert einen zuvor bereits angezeigten FileRequester. Ist der Handle Null oder wird kein Handel angegeben, werden die Verwaltungs-Strukturen neu angelegt. Bei Aufruf des FileRequester erhält man das File-Requester-Handle als Rückgabewert. Eine Null als Rückgabe bedeutet, daß ein Fehler bei der Anzeige des Requesters aufgetreten ist.

RF_Window
(struct Window *) - Adresse des Fensters in dem der File-Requester angezeigt werden soll. Falls RF_Screen nicht spezifiziert ist, wird die Adresse des Windows benutzt, um festzulegen, auf welchem Screen der File-Requester geöffnet wird.

RF_PubScreenName
(STRPTR) - Name eines Public-Screens, auf dem der File-Requester angezeigt werden soll.

RF_Screen
(struct Screen *) - Adresse des Screens, auf dem der File-Requester geöffnet werden soll. Diese Angabe überschreibt eine eventuell ebenfalls angegebene R_Window- oder R_PubScreenName-Tag.

RF_PrivateIDCMP
(BOOL) - Ist dieser Parameter auf TRUE gesetzt, wird ein neuer IDCMP-Port für das File-Requester-Window angefordert. Falls der Parameter nicht angegeben oder auf FALSE gesetzt ist, und der Parameter R_Window spezifiziert wurde, wird der IDCMP-Port der R_Window`s benutzt.

RF_IntuiFunc
(struct Hook *) - Mit diesem Parameter wird eine Funktion spezifiziert, zu der alle unbekannten Intuition-Nachrichten weitergeleitet werden, die beim IDCMPort des File-Requester Fensters ankommen. Die Funktion bekommt folgende Adressen geschickt:
A0 - (struct Hook *)
A1 - (struct FileRequester *)
A2 - (struct IntuiMessage *)

RF_SleepWindow
(BOOL) - Falls dieses Flag auf TRUE gesetzt ist, wird das Window, das durch die Adresse im TAG-Item R_Window spezifiziert wird, schlafen gelegt. Sobald man das Fenster aktiviert, wird ein Busy-Mouspointer angezeigt und keine Menü-Aktivitäten mehr zugelassen.

RF_UserData
(APTR) - Ein 32-Bit-Wert, der einfach in das fr_UserData Feld der Requester-Struktur kopiert wird.

RF_TextAttr
(struct TextAttr *) - Die Adresse einer Textattribut-Struktur, die festlegt, welcher Font im Window, in den Gadgets und in den Menüs verwendet werden soll. Ist dieser Tag-Item nicht angegeben oder dessen Wert auf NULL gesetzt wird der Default-Screen-Font des Screens benutzt, auf dem der Requester geöffnet wird.

RF_Locale
(struct Local *) - Dieser Parameter spezifiziert die Sprache, in der die Gadgets und Menüs des File-Requesters angezeigt werden sollen. Falls dieses Tag-Item nicht angegeben wurde, wird die im System voreingestellte Sprache genutzt. Nur ab AMIGA Workbench 2.1.

RF_TitleText
(STRPTR) - Der Titel, der im File-Requester Window angezeigt werden soll.

RF_PositivText
(STRPTR) - Der String, der im Positiv-Gadget des File-Requesters angezeigt werden soll. Default eingestellt ist "OK".

RF_NegativeText
(STRPTR) - Der String, der im Negativ-Gadget des File-Requesters angezeigt werden soll. Default eingestellt ist "Cancel".

RF_InitioalLeftEdge
(WORD) - Gewünschte X-Koordinate für das Requester-Window.

RF_InitialTopEdge
(WORD) - Gewünschte Y-Koordinate für das Requester-Window.

RF_InitialWidth
(WORD) - Gewünschte Breite des Requester-Windows.

RF_InitialHeight
(WORD) - Gewünschte Höhe des Requester-Windows.

RF_InitialFile
(STRPTR) - Eintrag, der beim Aufruf im File-Gadget des Requesters enthalten sein soll. Voreingestellt ist keine Eintragung.

RF_InitialDrawer
(STRPTR) - Eintrag, der beim Aufruf im Drawer-Gadget des Requesters enthalten sein soll. Der Inhalt des Verzeichnisses wird ebenfalls angezeigt. Voreingestellt ist das aktive Verzeichnis.

RF_InitialPattern
(STRPTR) - Eintrag, der bei Aufruf des Requesters im Pattern-Gadget enthalten sein soll. Voreingestellt ist #?.

RF_Flags1
(ULONG) - Gesamteinstellung vieler Optionen des Requesters.

RF_Flags2
(ULONG) - Weitere Gesamteinstellungen vieler Optionen des Requesters.

RF_DoSaveMode
(BOOL) - Dieses Flag sollte auf TRUE gesetzt werden, wenn der File-Requester zum Speichern benutzt wird.

RF_DoMultiSelect
(BOOL) - Dieses Flag sollte auf TRUE gesetzt werden, wenn der File-Requester zur Mehrfachauswahl benutzt werden soll. Diese Einstellung kann nicht im Speichermodus gemacht werden.

RF_DoPatterns
(BOOL) - Wenn ein Pattern-Gadget angezeigt werden soll, muß dieses Flag auf TRUE gesetzt werden.

RF_DrawersOnly
(BOOL) - Dieses Flag muß auf TRUE gesetzt werden, wenn nur Verzeichnisnamen und keine Dateinamen angezeigt werden sollen.

RF_FilterFunc
(struct Hook *) - Die Adresse einer Funktion, die für jede gefundeneDatei aufgerufen wird. Die Funktion liefert TRUE, wenn die Datei in der Dateiliste enthalten ist. Andernfalls wird die Datei nicht in der Anzeige mit ausgegeben. Die Funktion erwartet folgende Parameter:
A0 - (struct Hook *)
A1 - (struct FileRequester *)
A2 - (struct AnchorPath *)

RF_RejectIcons
(BOOL) - Dieses Flag sollte immer gesetzt sein, da dadurch die .Info-Dateien der Workbench-Icons nicht angezeigt werden.

RF_RejectPattern
(UBYTE *) - Über dieses Tag-Item kann ein AmigaDOS-Wildcard festgelegt werden, das zur Auswahl für Dateien, die nicht angezeigt werden sollen, genutzt wird. Das angegebene Wildcard muß zuvor von der Dos-Funktion ParsePatternNoCase() geprüft worden sein.

RF_AcceptPattern
(UBYTE *) - Über dieses Tag-Item kann ein AmigaDOS-Wildcard festgelegt werden, das zur Auswahl für Dateien, die angezeigt werden sollen, genutzt wird. Das angegebene Wildcard muß zuvor von der Dos-Funktion ParsePatternNoCase() geprüft worden sein.

RF_FilterDrawers
(BOOL) - Dieses Tag sollte auf TRUE gesetzt werden, wenn das R_RejectPattern, R_AcceptPattern und der Pattern-Text im Gadget auch auf Verzeichnisnamen angewandt werden soll. Normalerweise werden Verzeichnisnamen immer angezeigt.

RF_HookFunc
(APTR) - Eine Funktion, die aufgerufen wird, wenn R_FILTERFUNC oder R_INTUIFUNC im TAG-Item R_Flags1 spezifiziert wurde. Die Funktion sollte folgendermaßen aufgrufen werden können:

```
ULONG function( ULONG mask, APTR object, APTR x ( ?))
```

Der *mask*-Wert ist eine Kopie der R-Flags1 Einstellungen. *Object* ist eine Adresse auf

RF_OverWriteCheckMsg
(BOOL) - Bei der Auswahl einer Datei wird geprüft, ob diese bereits vorhanden ist und eine Abfragemeldung angezeigt, ob überschrieben werden soll oder nicht.

R_FilerCacheFree

void R_FilerCacheFree(appladdr, taglist);
A0 A1

APTR appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct TagItem *taglist; /* Lister der Einstellparameter */

Der Speicher, der durch einen File-Requester belegt wird, wird hierdurch wieder freigegeben.
Ist bei Aufruf des FileRequesters das Flag zum Zwischenspeichern (FRQCACHINGM) der Dateinamen eingestellt, muß spätestens beim Beenden des Programmes der Speicher wieder freigegeben werden.
Bei Aufruf der Funktion R_ExitAppl geschieht dies für alle währende des Programmlaufs angezeigten FileRequester.

Mögliche TagParameter:

RF_Handle
Das Hand eines zuvor aufgerufenen File-Requesters. Dieser Parameter kann in der Liste mehrmals angegeben werden.

R_FormAlert (V16)

id = R_FormAlert(appladdr, taglist);
D0 A0 A1

```
WORD        id;       /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct TagItem *taglist; /* Adresse der Eingabeparameterliste */
```

Mit dieser Funktion wird ein Fehlermeldungendialog angezeigt. Mit dem Parametern kann man sein Aussehen und seine Handhabung genauer eingestellen.
Die Anzahl und Breite der Textzeilen richtet sich nach den Trennsymbolen im Text. Ebenso die Anzahl der dargestellten Gadgets. Natürlich kann der Text nicht breiter als der Screen werden. In diesem Fall wird automatisch getrennt. Auch bei den Gadgets wird eine weitere Zeile eingefügt, wenn der im Gadget anzuzeigende Text zu lang wird.

Der String hat folgendes Format:

"[SYMBOLNUMMER][1. Textzeile|2. Textzeile|3. Text...|...][1. Gadget|2. Gadg...|...]"

Wie Sie sehen werden die Textzeile und die Gadgets durch ein |-Zeichen voneinander getrennt. Die erste Ziffer, die mit SYMBOLNUMMER bezeichnet ist, wählt das Symbol, das links im Requester erscheinen soll.

Bei der Symbolwahl haben Sie vier Möglichkeiten:

0-> kein Symbol
1-> NOTE; ein auf der Spitze stehendes Quadrat mit Ausrufezeichen
2-> WAIT; ein Fragezeichen innerhalb eines Achtung-Schildes
3-> STOP; ein Stop-Schild

Eines der Gadgets kann als Default Gadget definiert werden. Das Default Gadget ist durch eine andere Textfarbe unübersehbar markiert und kann direkt mit der Return/ Enter-Taste ausgewählt werden. Ist der Requester auf dem Bildschirm angezeigt, kann das Default-Gadget mit der Cursor-Links- oder mit der Tab-Taste auf das rechts daneben und mit der Cursor-Rechts- oder Shift/Tab-Taste auf das links daneben befindliche Gadget gewechselt werden. Es ist immer das Gadget mit Return wählbar, auf demdie hervorgehobene Textfarbe angezeigt wird.
Ist kein Default-Gadget angegeben, wird kein Rahmen gezeichnet und es ist auch nicht möglich, wie beschrieben, die Gadgets mit der Tastatur, zu wechseln.

Man hat folgende Einstellmöglichkeiten:

RFA_DefaultID
Nummer des Default-Gadgets.

RFA_AlertText
Alarmtext, mit im Folgenden beschriebenen Format . Der Parameter kann in der Liste wiederholt werden.

RFA_GadgetText
Der Text für die Gadgets. Die Gadgettexte können auch im FA_AlertText nach beschriebenem Format untergebracht sein. Der Parameter kann in der Liste wiederholt werden.

RFA_Timeout
Die Zeit in Sekunden, bis der Requester automatisch wieder vom Bildschirm verschwinden soll. Wenn nichts anderes eingestellt ist, wird die Default-Gadgetnummer zurückgeliefert.

RFA_DefaultTimeout
Legt die ID fest, die bei Timeout des Alarm-Requesters zurückgeliefert werden soll.

RFA_BeginSoundSample
Adresse eines IFF-Samples, das nach erscheinen des Requesters abgespielt werden soll.

RFA_EndSoundSample
Adresse eines IFF-Samples, das bei Beendung des Requesters abgespielt werden soll.

RFA_TimeoutSoundSample
Adresse eines IFF-Samples, das bei Timeout abgespielt werden soll.

RFA_OnlySoundSample
Es soll nur das angegebene Sample abgespielt werden. Der Requester wird nicht angezeigt.

RFA_BeginSpeek
Text der nach Aufruf des Requester gesprochen werden soll.

RFA_EndSpeek
Text der nach Beendung des Requesters gesprochen werden soll.

RFA_OnlySpeek
Es soll nur der angegebene Text gesprochen werden. Der Requester wird nicht angezeigt.

RFA_Human
Männliche-, weibliche- oder eine Roboterstimme. Betrifft nur die Sprachausgabe.

RFA_WindowFlags
Der Alert soll innerhalb eines Fensters mit den in ti_Data angegebenen Flags angezeigt werden.

RFA_WindowDrag
Das Fenster soll verschiebbar sein.

RFA_WindowDepth
Das Fenster soll mit einem Gadget ausgestattet sein, mit dem es in den Hintergrund geklickt und wieder hervorgeholt werden kann.

RFA_WindowClose
Das Fenster soll zusätzlich mit einem CloseGadget ausgestattet sein.

RFA_CloseID
Die zurückzugebende ID bei Anwahl des Schließfeldes des Fensters.

RFA_Image
Auswahl eines der vier möglichen Images:
0-> kein Symbol
1-> NOTE; ein auf der Spitze stehendes Quadrat mit Ausrufezeichen
2-> WAIT; ein Fragezeichen innerhalb eines Achtung-Schildes
3-> STOP; ein Stop-Schild

RFA_ImageData
Adresse einer fertig vorbereiteten Image-Struktur. Das angegebene Image wird anstatt der wählbaren Images angezeigt. Der Tag-Eintrag kann mehrmals angegeben werden um so eine Image-Animation erzeugen zu können.

RFA_ImageToggle
Zeit in Millisekunden, die vergehen soll, bis ein weiteres angegebenes Image angezeigt wird. Es müssen mindesten zwei Image-Daten Tags in der angegebenen Tag-Liste enthalten sein.

RFA_Window
Die Adresse eines Fensters in dem die Alert-Meldung erscheinen soll.

RFA_Center
Das Alert-Fenster wird zentriert auf dem Screen angezeigt.

Beispiel:

```
#include <exec/types.h>
#include <intuition/intuition.h>
#include <libraries/dosextens.h>
#include <functions.h>
#include <stdio.h>
#include <rct/rctdef.h>
#include <rct/rctpragma.h>
#include <rct/rcttagfuncs.h>

APTR ApplBase;
APTR RctBase;

void main( void)
{
if( ! (RctBase = (APTR)OpenLibrary( ( UBYTE *)"rct.library",16L)))
    {
    printf("Can't open libraries\n");
    return;
    }
if( ApplBase = R_InitApplTags( TAG_DONE))
    {   /* Ein normaler Alert */
    R_FormAlertTags( ApplBase, RFA_DefaultID, 1, RFA_Center, 1,
    RFA_AlertText, "[1][Hallo|dies ist ein ganz normaler Alert!][OK]",
    TAG_DONE);
        /* Der ist schon etwas chaotischer */
    R_FormAlertTags( ApplBase, RFA_WindowFlags,
    WINDOWCLOSE|WINDOWDEPTH,RFA_DefaultID, 1,
    RFA_GadgetText,"1. Gadget",RFA_AlertText,
    "[1][Hallo|dies ist ein ganz toller Alert!][2. _Gadget]",
    RFA_AlertText, "zeile3|zeile4|zeile5|zeile6",
    RFA_GadgetText,"3. Gadget", TAG_DONE);
    R_ExitAppl( ApplBase);
    }
CloseLibrary( ( struct Library *)RctBase);
}
```

Das Screen-Mode Auswahlfenster

R_ModeIDReq

ScrMdStruct = R_ModeIDReq(appladdr, taglist);
D0 A0 A1

ScrMdStruct = R_ModeIDReqTags(appladdr, tag, ...);

```
struct R_ScrModeReq  *ScrMdStruct;   /* Pointer auf struct R_ScrModeReq */
APTR                 appladdr;       /* von R_InitAppl zurückgelieferte Adresse
```

Durch Aufruf der Funktion wird das Auflösungsauswahlfenster angezeigt. Die darin angezeigten Auflösungen entsprechen immer den, im jeweiligen System tatsächlich vorhandenen, einstellbaren Auflösungen für Screens.

```
struct R_ScrModeReq
{
    ULONG  rs_DisplayID; /* Display Mode ID      */
    ULONG  rs_DisplayWidth; /* Display Weite in Pixel  */
    ULONG  rs_DisplayHeight;    /* Display Höhe in Pixel       */
    ULONG  rs_DisplayDepth; /* Anzahl der Bitplanes für das Display  */
    ULONG  rs_Overscantype; /* Overscantyp für das Display     */
    BOOL   rs_AutoScroll;   /* Ist Autoscroll möglich  */
    ULONG  rs_BitMapWidth;  /* Zum Erzeugen einer eigenen BitMap     */
    ULONG  rs_BitMapHeight;
    WORD   rs_LeftEdge;  /* Die Koordinaten für den Requester */
    WORD   rs_TopEdge;
    WORD   rs_Width;
    WORD   rs_Height;
    BOOL   rs_InfoOpened;
    WORD   rs_InfoLeftEdge;
    WORD   rs_InfoTopEdge;
    WORD   rs_InfoWidth;
    WORD   rs_InfoHeight;
    APTR   UserData; /* Für eigene Daten */
}
```

Die oben angegebene Struktur kann der Funktion zum Darstellen des ScreenMode Auswahlfensters übergeben werden damit die Größe und Position des Requesters die gleiche ist, wie bei einem eventuell zuvor gemachen Aufruf.

In der nachfolgenden Liste werden alle für das ScreenMode-Auswahlfenster nutzbare TagParameter aufgelistet:

RSM_InitialDisplayID

Die Display-ModeID, die beim Öffnen des Auswahlfensters voreingestellt ausgewählt sein soll.

RSM_DoWidth
RSM_InitialDisplayWidth
RSM_MinWidth
RSM_MaxWidth

Die beschriebenen Parameter legen fest, ob ein Schieberegler angezeigt werden soll, mit dem die Screen-Weite gewählt werden kann. Dazu kann die voreingestellt Weite übergebe und die minimale und maximale Weite festgelegt werden.

RSM_DoHeight
RSM_InitialDisplayHeight
RSM_MinHeight
RSM_MaxHeight

Durch Angabe dieser Parameter wird ein Schieberegler angezeigt, mit dem die Höhe es Screens wählbar ist. Angegeben wird die voreingestellte, die minimale und maximale Höhe für den wählbaren Screen.

RSM_DoDepth
RSM_InitialDisplayDepth
RSM_MinDepth
RSM_MaxDepth

Diese Parameter lassen einen Schieberegler erscheinen, mit dem die Bit-Tiefe des Screens eingestellbar ist. Angegeben wird die voreingestellte, die minimale und maximale Tiefe für den wählbaren Screen.

RSM_DoOverScanType
RSM_InitialOverScanType

Werden diese beiden Paramter angegeben, wird ein Blättersymbol dargestellt, mit dem die Overscanart gewählt werden kann. In **RSM_InitialOverscanType** wird der voreingestellte Typ übergeben.

RSM_DoAutoScroll

RSM_InitialAutoScroll

Mit diesen Parametern wird eine Auswahlfeld für den Screen-AutoScroll angezeigt. In **RSM_InitialAutoScroll** wird der voreingestellte Wert übergeben.

RSM_InitialLeftEdge

RSM_InitialTopEdge

Diese zwei Parameter bestimmen die Position an der das Fenster angezeigt werden soll.

RSM_UseOldValues

In „ti_Data" wird die Adresse einer „R_ScrModeReq-Struktur" erwarte, aus der die Werte für die Position des Fensters übernommen werden.

Lokalisierung und die Auswahlmöglichkeit von Gadgets per Tastatur.

Um eigene Software auf Märkten im In- und Ausland verkaufen zu können, ist eine Anpassung auf die jeweilige Landessprache unerläßlich. Erst mit neueren Betriebssystemversionen ist dies einigermaßen komfortabel möglich. Die Möglichkeiten, die Ihnen mit der Ressource-Bibliothek geboten werden sind mit Betriebssystemversion 1.3 genauso einsetzbar wie unter den neuesten Versionen.

Gleichzeitig mit der Oberflächen-Datei kann eine ASCII-Datei geladen werden, in der die alternativen Strings im EBN-Format vorliegen. Da es sich um eine ASCII-Datei handelt, kann man diese sehr einfach in eigene Dateien, in denen beispielsweise die Einstellungen des Programmes gespeichert werden, einbeziehen. Die Routine zum Auswerten der Datei, liest nur die Abschnitte, die durch die jeweiligen Schlüsselworte eingeleitet werden.

Die Datei ist folgendermaßen aufgebaut:

Nach dem Schlüsselwort **R_LOCALE_SECTION** folgt die genaue Angabe des Aufbaues der nachfolgenden Übersetzungs-Section.

Beispiel:

```
R_LOCALE_SECTION
```

Durch das Schlüsselwort **R_LOCALE_ISTRINGS** wird mitgeteilt, daß die nachfolgenden Texte bis zum nächsten Schlüsselwort zur Übersetzung der in MakeAPP für IntuiTexte angegebenen Strings eingesetzt werden soll.

Beispiel:

```
R_LOCALE_ISTRINGS
      "_ENDE",    RS_CHANGE,        CONTROL
      "_Abbruch", RS_CHANGE,        CONTROL
      "_OK",      RS_CHANGE,        CONTROL
```

Den Übersetzungsstrings folgt das Darstellungsflag und der Qualifier der Sondertaste, die in Verbindung mit dem nach dem Unterstrich angegebenen ASCII-Zeichen die Bestätigung auslösen soll. Durch Angabe von zwei Unterstrichen wird das Shortcut ignoriert und das normale ASCII-Zeichen als Unterstrich gezeichnet.

Mögliche Darstellungsflags (Bits):

NOCHANGE
keine Veränderung am Text vornehmen. Die angegebenen Strings werden ignoriert.

RS_CHANGE
Text soll eingesetzt werden.

RS_LJUSTIFY
Text bei Größenunterschied nach links ausrichten.

RS_RJUSTIFY
Bei Größenunterschied, Text nach rechts ausrichten. (Werden beide Flags angegeben, wird der Text zur ursprünglichen Position zentriert.)

RS_BORDERJUSTIFY
Bei Gadgets soll der Rahmen mitangepasst werden.

Die Flags für Links-, Rechtsausrichtung oder Borderanpassung haben natürlich bei Menütexten keinerlei auswirkung. Menüs lassen sich sehr einfach durch Zuhilfenahme der Funktion R_MenuDraw automatisch justieren!

Mögliche Qualifier:

NOQUALIFIER
Nachfolgende ASCII Zeichen werden direkt ausgewertet. Die Taste muß nicht in Verbindung mit einer Sondertaste gedrückt werden.

RIEQ_LSHIFT
Linke Shift-Taste.

RIEQ_RSHIFT
Rechte Shift-Taste.

RIEQ_CONTROL
Control-Taste.

RIEQ_LALT
Linke Alternate-Taste.

RIEQ_RALT
Rechte Alternate-Taste.

RIEQ_LCOMMAND
Linke AMIGA-Taste.

RIEQ_RCOMMAND
Rechte AMIGA-Taste.

Bei einer „Qualifier-Angabe"wie

RIEQ_LSHIFT|RIEQ_RSHIFT|RIEQ_CONTROL|RIEQ_LALT|RIEQ_RALT

wird erwartet, daß einige Sondertasten gedrückt sind, damit eine entsprechende Nachricht geschickt werden kann.

Dem Schlüsselwort **R_LOCALE_NSTRINGS** folgen sonstige Strings, die nicht direkt in Requester und Menüleisten Verwendnung finden, sondern als Fenstertitel oder Alert-Nachricht eingesetzt werden.

Beispiel:

R_LOCALE_NSTRINGS
RMSG_PROGENDE /* Identifier */
"Programm beenden?",

RMSG_DELFILE /* Identifier */
"Datei löschen?",

Der anzugebende Identifier wird als Konstantendefinition in der ausgewählten Programmiersprache abgespeichert. Der Identifier sollte immer das Prefix **RMSG_** beinhalten, damit Sie in Ihren Programmen eine eindeutige Zuweisung machen können.
Wie Sie im letzten Beispiel sehr gut sehen können, kann die Datei sogar mehrzeilige Kommentare enthalten. Ein Kommentar wird durch ein „/*" eingeleitet und mit einem „*/" beendet. Für C-Programmierer ist es sicher keine Umgewöhnung.

Beispiel in C:

Folgende Konstanten-Datei muß angelegt werden:

```
enum RMSGLOC_DEFINES
{
RMSG_PROGENDE,
RMSG_DELFILE
}
```

C-Programm:

```
void main( void)
{...
printf( "%s\n",R_GetLocStr( appladdr, RMSG_PROGENDE);
...}
```

R_LOCALE_END
Diese Schlüsselwort markiert das Ende der Locale-Section.

Tastaturkürzel (Short-Cuts)

Wie allgemein üblich, wird die Tastaturkürzeldefinition im Text des jeweiligen Gadgets selbst vorgenommen. Ein Unterstrich vor einem Zeichen (z.B. _Ja) bedeutet, daß bei der Anzeige des Textes der nachfolgende Buchstabe unterstrichen (_Ja wird zu Ja) und der ASCII-Wert des entsprechenden Zeichens als Auswahlkürzel genutzt werden soll. Ein doppelter Unterstrich (z.B. __Ja) bedeutet, daß kein Kürzel und damit verbunden kein Unterstrich unter den Buchstaben, sondern tatsächlich nur ein Unterstrich vor den Buchstaben gesetzt wird(__Ja wird zu _Ja). Die Bibliotheks-Funktionen R_GetMsg und R_ReplyMsg filtern die Nachrichten automatisch für Sie aus, so daß Sie in Ihrem Programm zum Beispiel nur auf Gadget-Nachrichten warten müssen, und diese auch bekommen obwohl die Tastatur bedient wurde.

R_ResLoad (V16)

handle= R_ResLoad(appladdr, taglist);
D0 A0 A1

```
LONG            handle;   /* Der für R_ChangeAccess benötigte Dateihandle */
APTR            appladdr; /* Rückgabewert von R_InitAppl */
struct TagItem  *taglist; /* Liste der Übergabeparameter */
```

Die angegebene Ressource-Datei wird geladen und ein Handle zurückgeliefert. Bei Angabe von Dateinamen, die auf der Festplatte oder Diskette vorhanden sind, ist es sinnvoll nur den Namen ohne Pfadbeschreibung anzugeben. Geben Sie nur den Dateinamen an, wird zuerst im aktuellen Verzeichnis und dann im Verzeichnis mit dem Namen RCT: gesucht. Schlägt die Suche fehl, wird das Dateiauswahlfenster angezeigt und Sie können die RCT-Datei selbst lokalisieren.

Änderungen/Neuerungen:
In den Requester-Flags wird das Flag NOISYREQ generell gesetzt. Dadurch besteht die Möglichkeit, Rawkey-Codes auch bei angezeigtem Requester zu empfangen. Benutzen Sie zum Empfangen von Nachrichten die Funktion *R_GetMsg*.

Erweiterung:
Hinzugekommen ist die Möglichkeit, beliebig viele Ressource-Dateien in den Speicher zu laden. Sie müssen sich dabei lediglich den Dateinamen merken, da der Zugriff über den Dateinamen freigegeben wird. Die zuletzt geladene Datei wird als aktuell angesehen und hat immer Zugriffsvorrecht.

> **Hinweis:** Die Funktion *R_OldRctLoad* ist mit dieser zwar weitgehend identisch, sollte aber im Zusammenhang mit den Funktionen *R_InitAppl* und *R_ExitAppl* nicht mehr benutzt werden.

Tag-Parameter:

Rres_Name
Die Adresse des Dateinamens.

Rres_Address
Adresse einer Resource-Datei, die sich bereits im Speicher befindet. Man hat dadurch die Möglichkeit, die Datei direkt an das Programm zu binden und muß nach dem Start des Programmes lediglich die Adresse des Speicherbereiches an die Funktion übergeben.

Achtung: Da die Funkion die angebundene Datei relociert, kann die Funktion kein zweites Mal auf denselben Speicherbereich angewendet werden. Für einen Neustart sind immer nicht relocierte Daten erforderlich. Das Programm muß also neu geladen, compiliert oder assembliert werden.

Rres_LocalPath

Die Adresse eines Dateinamens einer mit dem externen Programm **StripRes** oder mit MakeAPP generierten Datei. Die Datei beinhaltet alternative Textstrings für IntuiTexte. Die Reihenfolge der Textstrings entspricht strickt der, der Intui-Textstrings in der Ressource-Datei.

Achtung: Stimmt die Anzahl der geladenen Strings nicht mit denen der Intui-Textstrings überein, hat das Laden der Datei keinerlei Auswirkung auf die Resource. Der belegte Speicher wird dann wieder freigegeben.

Rres_LocalFlags

Die durch diesen Parameter angegebenen Flags beeinflussen die Datei mit den alternativen Textstrings.

0 -> (default) Die Strings werden direkt nach dem Laden der Dateien eingesetzt.
1 -> Erst nach Aufrufen der Funktion ***R_Localize*** sollen die Strings eingesetzt werden.

Verweis: R_ChangeAccess, R_ResFree, R_ExitAppl,R_Localize

Beispiel:

```
#include <rct/rctdef.h>
#include <rct/rcttagfuncs.h>

APTR ApplAddr;
char pfad[] = {"RCTDEMO.RCT"};

void main( void)
{
... /* Libraries öffnen */

if( ApplAddr = R_InitApplTags( TAG_DONE))
  {
  if( R_ResLoadTags( ApplAddr, Rres_Name, pfad, TAG_DONE))
    {
    ...
    /* hier geht es weiter */
    }
  /* Resourcen freigeben */
  R_ExitAppl( ApplAddr);
  }
else
  ... /* Initialisierung fehlgeschlagen */
... /* Libraries schließen */
}
```

R_ResFree

error = R_ResFree(appladdr);
D0 A0

```
WORD  error,     /* Rückgabewert */
APTR  appladdr;  /* Rückgabewert der Funktion R_InitAppl */
```

Gibt den durch die Funktionen *R_OldRctLoad* oder *R_ResLoad* reservierten Speicherplatz wieder frei. Die Funktion sollte vor dem Ende jeden Programmes aufgerufen werden.
Wurde die Funktion ausgeführt, kann jeder weitere Zugriff auf die freigegebene Datei zu einem Systemabsturz führen.

> **Hinweis:** Werden die erweiterten Funktionen genutzt, wird entweder die zuletzt geladene Datei oder die zuletzt mit der Funktion *R_ChangeAccess* aktivierte Datei aus dem Speicher gelöscht.

Beim Aufruf der Funktion R_ExitAppl werden alle geladenen Ressource-Dateien aus dem Speicher gelöscht. Es ist also mit der Funktion R_ResFree möglich, geziehlt Dateien aus dem Speicher zu entfernen.

R_ChangeAccess (V16)

R_ChangeAccessTags (V16)

error = R_Changeaccess(appladdr, taglist)
D0 A0 A1

error = R_Changeaccesstags(appladdr, tag, ...)

```
WORD          error;     /* Rückgabewert */
APTR          appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct TagItem *taglist; /* Adresse einer Tag-Liste */
oder
ULONG         tag, ...;  /* Die Tag-Parameter und Werte direkt */
```

Mit dieser Funktion kann der Zugriff auf eine bestimmte bereits im Speicher befindliche Ressource-Datei eingestellt werden. Die Datei muß zuvor mit der Funktion *R_ResLoad* geladen worden sein.

Die Tag-Parameter der Funktion:

Rres_Name
Als Parameter wird der Name der Datei übergeben, die geladen wurde.

Rres_Handle
Der Parameter ist der Handle, den die Funktion R_ResLoad zurückliefert.

Beispiel:

```
#include <rct/rctdef.h>
#include <rct/rcttagfuncs.h>

#define REQONE 0
#define REQTWO 0

char firstdatei[] = {"RCTDEMO1.RCT};
char seconddatei[] = {"RCTDEMO2.RCT};
APTR ApplAddr;

void main( void)
{
struct Requester *firstreq, *secreq;
... /* Libraries öffnen! */
```

```
if( ApplAddr = R_InitApplTags( TAG_DONE))
  {
  if( R_ResLoadTags( ApplAddr, Rres_Name, firstdatei, TAG_DONE))
    {
    if( R_ResLoadTags(  ApplAddr, Rres_Name, seconddatei, TAG_DONE))
      {
      /* liefert die Adresse des ersten Requesters der zuletzt
        geladenen Datei */
      secreq = ( struct Requester *) R_ResAddr( ApplAddr, 0, REQTWO);
      if( R_Changeaccesstags( ApplAddr, Rres_Name, firstdatei))
        {
        /* liefert die Adresse des ersten Requester der zuerst
          geladgenen Datei */
        firstreq = ( struct Requester *) R_ResAddr(ApplAddr, 0, REQONE);
        }
      }
    ...
    }
  ...
  }
...
}
```

R_ReqCenter

error = R_ReqCenter(ApplAddr, req, window)
D0 A0 A1 A2

```
WORD              error;r     /* Rückgabewert */
APTR              appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;       /* die mit der Funktion R_ResAddr() ermittelte
                                 Adresse einer Requester-Struktur*/
oder
ULONG             req;        /* Requester-ID */
struct Window     *window;    /* Adresse des Windows, innerhalb dessen
                                 der Requester zentriert werden soll */
```

Zentriert den angegebenen Requester innerhalb der Ausmaße des angegebenen Fensters. Ist der Requester bereits angezeigt, hat dies keine Auswirkungen auf die Position des Requesters. Erst bei erneutem Anzeigen wird die Position neu gesetzt. Diese Funktion sollte deshalb vor dem Zeichnen des Requesters aufgerufen werden.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_ReqDraw

error = R_ReqDraw(appladdr, req, win)
D0 A0 A1 A2

```
SHORT             error;      /* Rückgabewert */
APTR              appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;       /* die mit der Funktion R_ResAddr() ermittelte
                                 Adresse einer Requester-Struktur*/
oder
ULONG             req;        /* Requester-ID */
struct Window     *window;    /* Adresse des Windows, innerhalb dessen der Requester
                                 zentriert werden soll */
```

Zeichnet den angegebenen Requester in das angegebene Fenster. Ist nicht genügend Speicherplatz vorhanden, wird NULL zurückgeliefert, andernfalls 1.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_ReqDrawWindow

window = R_ReqDrawWindow(appldr, req, screen, titel)
D0 A0 A1 A2 A3

```
struct Window      *window; /* Rückgabewert */
APTR               appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester   *req;    /* die mit der Funktion R_ResAddr() ermittelte
                               Adresse einer Requester-Struktur*/
```

oder

```
ULONG              req;     /* Requester-ID */
struct Screen      *screen; /* Adresse des Screens, innerhalb dessen das Fenster
                               angezeigt werden soll */
```

Zeichnet den angegebenen Requester und öffnet zuvor ein neues Fenster. Das Fenster umschließt den Requester komplett, ist verschiebbar und kann in den Vorder- oder Hintergrund geklickt werden. Ohne Angabe eines Screens wird das Fenster auf dem Screen geöffnet, der für Systemmeldungen aktivierte wurde (siehe Funktionen R_SetSysRequest( ), R_ResetSysRequest( )).

Konnte die Funktion erfolgreich ausgeführt werden, wird die Adresse des Fensters zurückgeliefert, andernfalls Null.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GadDrawWindow

window = R_GadDrawWindow(appladdr, req, scr, title, idcmp, flags);
D0 A0 A1 A2 A3 D0 D1

```
struct Window    *window; /* Rückgabewert */
APTR             appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;     /* Adresse einer Requester-Struktur */
oder
LON              req;      /* Requester-ID */
struct Screen    *scr;     /* Adresse einer Screen Struktur */
BYT              *title;   /* Der Text für das angegebene Fenster */
ULONG            idcmp;    /* Die Fenster IDCMP-Flags */
ULON             flags;    /* Die Flags für das Fenster */
```

Öffnet ein Fenster in der Größe des angegebenen Requesters und zeigt diesen darin an. Ohne Angabe einer Screenadresse wird das Fenster auf dem Workbenchscreen geöffnet. Beim Schließen des Requesters mit der Funktion *R_ReqEnd* wird auch das Window geschlossen. Der einzige Unterschied zu der Funktion reqdrawwindow ist der, daß alle Gadgets, Requestertexte und Border direkt mit dem Window verknüpft werden. Man hat dadurch den Vorteil, daß zusätzlich zur Abfrage der Gadgets noch die Menüleiste berücksichtigt werden kann.

Die Erklärung, zu den einstellbaren IDCMP- und Fenster-Flags finden Sie bei der Erleuterung zur Funktion R_ShowWindow.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_ReqEnd

error = R_ReqEnd(appladdr, req, win);
D0 A0 A1 A2

```
SHORT           error;      /* Rückgabewert */
APTR            appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;      /* die mit der Funktion R_ResAddr() ermittelte
                               Adresse einer Requester-Struktur*/
```

oder

```
ULONG           req;        /* Requester-ID */
struct Window   *win;       /* Adresse des Fensters, innerhalb dem der Requester
                               angezeigt wird */
```

Der in *req* angegebenen und auf dem Bildschirm angezeigte Requester wird vom Bildschirm gelöscht.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_ReqWaitMsg

gadgetid = R_ReqWaitMsg(appladdr, req, &messageart);
D0 A0 A1 A2

```
SHORT           gadgetid;  /* Rückgabewert */
APTR            appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;     /* die mit der Funktion R_ResAddr() ermittelte
                              Adresse einer Requester-Struktur*/
oder
ULONG           req;       /* Requester-ID */
ULONG           Message;   /* Adresse einer 4-Byte-Variablen */
```

Wartet auf Nachrichten, die auftreten, wenn mit der Maus Gadgets, innerhalb des angegebenen Requesters, angeklickt werden. Zurückgeliefert wird dann die ID-Nummer des angeklickten Gadgets und in der Variablen *Message*, ob das Gadget gedrückt (GADGETDOWN) oder losgelassen wurde (GADGETUP).

Da diese Funktion nur die Gadget-Nachrichten liefert, ist sie nur für einfache Aufgaben geeignet. Benutzen Sie besser die Funktion *R_GetMsg*.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GetReqStruct

wert = R_GetReqStruct(appladdr, req, item);
D0 A0 A1 D0

```
void              *wert;    /* Rückgabewert, je nach Eintrag
                            verschieden lang */
APTR              appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;     /* die mit der Funktion R_ResAddr() ermittelte Adresse
                            einer Requester-Struktur*/
oder
ULONG             req;      /* Requester-ID */
SHORT             item;     /* Strukturteil, dessen Wert ermittelt werden soll */
```

Liefert den Wert eines Strukturteils einer Requester-Struktur.
Folgende Werte kann *item* annehmen:

GRS_OLDERREQUEST -> struct Requester *OlderRequest
GRS_LEFTEDGE -> LeftEdge
GRS_TOPEDGE -> TopEdge
GRS_WIDTH -> Width
GRS_HEIGHT -> Height
GRS_RELLEFT -> RelLeft
GRS_RELTOP -> RelTop
GRS_REQGADGET -> struct Gadget *ReqGadget
GRS_REQBORDER -> struct Border *ReqBorder
GRS_REQTEXT -> struct IntuiText *ReqText
GRS_FLAGS-> Flags
GRS_BACKFILL -> BackFill
GRS_REQLAYER -> struct Layer *ReqLayer
GRS_IMAGEBMAP -> struct BitMap *ImageBMap
GRS_WINDOW -> struct Window *RWindow

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetReqStruct

error = R_SetReqStruct(appladdr, req, item,wert)
D0 A0 A1 D0 D1

```
SHORT           error;   /* Rückgabewert */
APTR            appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;   /* die mit der Funktion R_ResAddr() ermittelte Adresse
                            einer Requester-Struktur*/
oder
ULONG           req;     /* Requester-ID */
SHORT           item;    /* Strukturteil, dessen Wert gesetzt werden soll */
SHORT           wert;    /* neuer Wert für den angegebenen Strukturteil */
```

Setzt den in *item* angegebenen Strukturteil der Requester-Struktur auf den in *wert* angegebenen Wert.

Folgende Werte kann *item* annehmen:

SRS_LEFTEDGE	-> LeftEdge
SRS_TOPEDGE	-> TopEdge
SRS_WIDTH	-> Width
SRS_HEIGHT	-> Height
SRS_RELLEFT	-> RelLeft
SRS_RELTOP	-> RelTop
SRS_FLAGS	-> Flags
SRS_BACKFILL	-> BackFill

Hinweis:
Ist der Requester angezeigt, verändert sich nichts an seinem Aussehen. Erst wenn er erneut angezeigt wird, wird eine Veränderung sichtbar.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetReqText

error = R_SetReqText(appladdr, req, id, stringadr)
D0 A0 A1 D0 A2

```
SHORT           error;      /* Rückgabewert */
APTR            appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;      /* die mit der Funktion R_ResAddr() ermittelte
Adresse einer Requester-Struktur*/
oder
ULONG           req;        /* Requester-ID */
WORD            id;         /* Objektnummer des Requestertextes */
UBYTE           *stringadr; /* Die Adresse des Strings. */
```

Setzt einen neuen Textstring in einen Requestertext. Ist der Requester angezeigt, wird das Textfeld neu gezeichnet, so daß der neue Text direkt sichtbar wird.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requester übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeApp definiert wurde.

R_GetReqText

stringaddr = R_GetReqText(appladdr, req, id)
D0 A0 A1 D0

```
UBYTE           *stringaddr;    /* Rückgabewert */
APTR            appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;     /* die mit der Funktion R_ReaAddr() ermittelte Adresse
                            einer Requester-Struktur*/
oder
ULONG           req;        /* Requester-ID */
WORD            id;         /* Objektnummer des Requestertextes */
```

Liefert die Adresse des Textstrings des angegebenen Requestertextes.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetGadText

error = R_SetGadText(appladdr, req, id, stringadr)
D0 A0 A1 D0 A2

```
SHORT              error;     /* Rückgabewert */
APTR               appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester   *req;      /* die mit der Funktion R_ResAddr() ermittelte Adresse
                                 einer Requester-Struktur*/
oder
ULONG              req;       /* Requester-ID */
WORD               id;        /* Objektnummer des Gadgets */
UBYTE              *stringadr; */ die Adresse eines „nullterminierten" Strings */
```

Setzt anstelle des Gadgettextes einen neuen Textstring ein. Ist der Requester auf dem Bildschirm angezeigt, wird das Gadget gleichzeitig neu gezeichnet, so daß der neue Text direkt sichtbar wird.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GetGadText

stringadr = R_GetGadText(appladdr, req, id)
D0 A0 A1 D0

```
UBYTE              *stringadr; /* Rückgabewert */
APTR               appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester   *req;      /* die mit der Funktion R_ResAddr() ermittelte Adresse
                                 einer Requester-Struktur*/
oder
ULONG              req;       /* Requester-ID */
WORD               id;        /* Objektnummer des Gadgets */
```

Ermittelt die Adresse des Gadgettexts, der im Gadget mit der Objektnummer *id* gesetzt ist.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GetGadStruct

wert = R_GetGadStruct(appladdr, req, id, item)
D0 A0 A1 D0 D1

```
void                *wert;     /* Rückgabewert */
APTR                appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester    *req;      /* die mit der Funktion R_ResAddr() ermittelte Adresse
                                  einer Requester-Struktur*/
oder
ULONG               req;       /* Requester-ID */
WORD                id;        /* Objektnummer des Gadgets */
WORD                item;      /* Strukturteil, dessen Wert ermittelt werden soll */
```

Ermittelt den Wert des Strukturteils *item* der Gadget Struktur des in der Objektnummer *id* angegebenen Gadgets.

item kann folgende Werte annehmen:

GGS_STRUCT	-> Adresse der Gadget-Struktur
GGS_NEXTGADGET	-> struct Gadget *NextGadget
GGS_LEFTEDGE	-> LeftEdge
GGS_TOPEDGE	-> TopEdge
GGS_WIDTH	-> Width
GGS_HEIGHT	-> Height
GGS_FLAGS	-> Flags
GGS_ACTIVATION	-> Activation
GGS_GADGETTYPE	-> GadgetType
GGS_GADGETRENDER	-> APTR GadgetRender
GGS_SELECTRENDER	-> APTR SelectRender
GGS_GADGETTEXT	-> struct IntuiText *GadgetText
GGS_MUTUALEXCLUDE	-> MutualExclude
GGS_SPECIALINFO	-> APTR SpecialInfo
GGS_GADGETIS	-> Gadgetid
GGS_USERDATA	-> APTR UserData

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetGadStruct

error = R_SetGadStruct(appladdr, req, id, item, wert)
D0 A0 A1 D0 D1 D2

```
WORD            error;     /* Rückgabewert */
APTR            appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;     /* die mit der Funktion R_ResAddr() ermittelte Adresse
                              einer Requester-Struktur*/
oder
ULONG           req;       /* Requester-ID */
WORD            id;        /* Objektnummer des Gadgets */
WORD            item;      /* Strukturteil, dessen Wert gesetzt werden soll */
ULONG           wert;      /* neuer Wert für den Strukturteil */
```

Schreibt in das in *item* angegebene Strukturteil der Gadget-Struktur den Wert der Variablen *wert*.
Item kann folgende Werte annehmen:

SGS_LEFTEDGE -> LeftEdge
SGS_TOPEDGE -> TopEdge
SGS_WIDTH -> Width
SGS_HEIGHT -> Height

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_RedrawGad

error = R_RedrawGad(appladdr, req, id, anzahl)
D0 A0 A1 D0 D1

```
WORD              error;    /* Rückgabewert */
APTR              appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;     /* die mit der Funktion R_ResAddr() ermittelte Adresse
                               einer Requester-Struktur*/
oder
ULONG             req;      /* Requester-ID */
WORD              id;       /* Objektnummer des Gadgets */
WORD              anzahl;   /* Anzahl der neu zu zeichnenden Gadgets.*/
```

Zeichnet die in *anzahl* angegebenen Gadgets ab dem in *id* angegebenen Gadget neu. Dabei werden auch Füllungen erneuert und Unterstreichnungen gezeichnet.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_RedrawGadgets

error = R_RedrawGad(appladdr, req, id)
D0 A0 A1 D0

```
WORD              error;    /* Rückgabewert */
APTR              appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester  *req;     /* die mit der Funktion R_ResAddr() ermittelte Adresse
                               einer Requester-Struktur*/
oder
ULONG             req;      /* Requester-ID */
WORD              id;       /* Objektnummer des Gadgets */
```

Zeichnet alle in der Liste enthaltenen Gadgets, ab dem in *id* angegebenen Gadget neu. Dabei werden auch Füllungen erneuert und Unterstreichnungen gezeichnet.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_ActiveGad

error = R_ActiveGad(appladdr, req, id)
D0 A0 A1 D0

```
WORD            error;    /* Rückgabewert */
APTR            appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;    /* die mit der Funktion R_ResAddr() ermittelte Adresse
                             einer Requester-Struktur*/
oder
ULONG           req;      /* Requester-ID */
WORD            id;       /* Objektnummer des Gadgets */
```

Aktiviert das in *id* angegebene String-Gadget, so daß ohne vorheriges Anklicken mit der Maus eingegeben werden kann.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetStrGadText

error = R_SetStrGadText(appladdr, req, id, stringadr)
D0 A0 A1 D0 A2

```
WORD            error;    /* Rückgabewert */
APTR            appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;    /* die mit der Funktion R_ResAddr() ermittelte Adresse
                             einer Requester-Struktur*/
oder
ULONG           req;      /* Requester-ID */
WORD            id;       /* Objektnummer des Gadgets */
UBYTE          *stringadr;/* die Adresse des Strings */
```

Setzt einen neuen Text in das Eingabefeld eines String Gadgets. Bei angezeigtem Requester wird der neue Inhalt direkt angezeigt.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GetStrGadText

stringadr = R_GetStrGadText(appladdr, req, id)
D0 A0 A1 D0

```
UBYTE           *stringadr; /* Rückgabewert */
APTR            appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;      /* die mit der Funktion R_ResAddr() ermittelte Adresse
                               einer Requester-Struktur*/
oder
ULONG           req;        /* Requester-ID */
WORD            id;         /* Objektnummer des Gadgets */
```

Ermittelt die Adresse des Textes, der in das Eingabefeld des String Gadget mit der Objektnummer *id* eingegeben wurde.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Requesters übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_GetStrGadStruct

wert = R_GetStrGadStruct(appladdr, req, id, item)
D0 A0 A1 D0 D1

```
WORD            wert;     /* Rückgabewert */
APTR            appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;    /* die mit der Funktion R_ResAddr() ermittelte Adresse
                             einer Requester-Struktur*/
oder
ULONG           req;      /* Requester-ID */
WORD            id;       /* Objektnummer des Gadgets */
WORD            item;     /* Strukturteil, dessen Wert ermittelt werden soll. */
```

Ermittelt den in *item* angegebenen Strukturteil der Stringinfo Struktur des String-Gadgets mit der Objektnummer *id*.

item kann folgende Werte annehmen:

GSGS_CWITH	-> CWidth
GSGS_HEIGHT	-> Height
GSGS_HPOTRES	-> HPotRes
GSGS_VPOTRES	-> VPotRes
GSGS_LEFTBORDER	-> LeftBorder
GSGS_TOPBORDER	-> TopBorder

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

R_SetPropGadStruct

error = R_SetPropGadStruct(appladdr, req, id, item, wert)
D0 A0 A1 D0 D1 D2

```
WORD            error;    /* Rückgabewert */
APTR a          ppladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Requester *req;    /* die mit der Funktion R_ResAddr() ermittelte Adresse
                             einer Requester-Struktur*/
oder
ULONG            req;     /* Requester-ID */
WORD             id;      /* Objektnummer des Gadgets */
WORD             item;    /* Strukturteil, dessen Wert gesetzt werden soll */
UWORD            wert;    /* neuer Wert für den Strukturteil */
```

Schreibt den in *wert* angegebenen Wert in den in *item* angegegeben Strukturteil der Propinfo-Struktur des Proportional Gagets mit der Objektnummer *id*.

item kann folgende Werte annehmen:

SPGS_FLAGS	-> Flags
SPGS_HORIZPOT	-> HorizPot
SPGS_VERTPOT	-> VertPot
SPGS_HORIZBODY	-> HorizBody
SPGS_VERTBODY	-> VertBody

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des RequesterS übergeben werden sondern es genügt die Übergabe der Requester-ID, wie sie in MakeAPP definiert wurde.

Menü-Funktionen

R_MenuDraw (V16)

R_MenuDrawTags (V16)

error = R_MenuDraw(appladdr, menu, window, taglist)
D0 A0 A1 A2 A3

error = R_MenuDrawTags(appladdr, menu, window, tag1, ...)

```
WORD          error;    /* Rückgabewert */
APTR          appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu   *menu;    /* Die Adresse der Menü-Struktur */
oder
ULONG         menu;     /* Menu-ID */
struct Window *window;  /* die Adresse des Menüfensters */
struct TagItem *taglist; /* Die Adresse der Parameterliste */
```

Setzt in das in *window* angegebene Window die in *menu* angegebene Menüleiste.

TagItems:

RMD_Justify
Das Menü wird automatisch an den Font angepasst, der für den Screen eingestellt ist, auf dem das Menüfenster steht.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_MenuEnd

error = R_MenuEnd(appladdr, menu, window)
D0 A0 A1 A2

```
WORD e      rror;     /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu *menu;    /* Die Adresse der Menü-Struktur */
oder
ULONG       menu;     /* Menu-ID */
struct Window *window; /* die Adresse des Menüfensters */
```

Löscht die in *menu* angegeben Menüleiste aus dem in *window* angegebenen Window. Die Menüleiste wird dabei nicht aus dem Speicher gelöscht. Dies geschieht mit der Funktion **R_ResFree()**.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_SetMenuTitle

error = R_SetMenuTitle(appladdr, menu, menuid, stringadr)
D0 A0 A1 D0 A2

```
WORD        error;    /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu *menu;    /* Die Adresse der Menü-Struktur */
oder
ULONG       menu;     /* Menu-ID */
WORD        menuid;   /* Objektnummer des Menüs */
UBYTE       *stringadr;/* die Adresse des Strings */
```

Schreibt den String, der an der Adresse *stringadr* im Speicher steht, in den Menüeintrag mit der Objektnummer *menuid*.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_GetMenuIText

stringadr = R_GetMenuItext(appladdr, menu, menuid, itemid)
D0 A0 A1 D0 D1

```
UBYTE          *stringadr; /* Rückgabewert */
APTR           appladdr;   /* Rückgabewert der Funktion R_InitAppl */
struct Menu    *menu;      /* Die Adresse der Menü-Struktur */
oder
ULONG          menu;       /* Menu-ID */
WORD           menuid;     /* Objektnummer des Menüs */
WORD           itemid;     /* Objektnummer des Menüitems */
```

Ermittelt die Adresse des Textstrings des Menüeintrages mit der Objektnummer *itemid* im Pull-Down Menü mit der Objektnummer *menuid*.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_SetMenuItemStruct

error = R_SetMenuItemStruct(appladdr, menu, menuid, itemid, item, wert)
D0 A0 A1 D0 D1 D2 D3

```
WORD        error;   /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu *menu;   /* Die Adresse der Menü-Struktur */
oder
ULONG       menu;    /* Menu-ID */
WORD        menuid;  /* Objektnummer des Menüs */
WORD        itemid;  /* Objektnummer des Menüeintrages */
WORD        item;    /* zu verändernde Strukturkomponente */
ULONG       wert;    /* neuer Wert für die Strukturkomponente */
```

Schreibt den in *wert* enthaltenen Wert in die Strukturkomponente *item* des Menüeintrages mit der Objektnummer *itemid* im Pull-Down-Menü mit der Objektnummer *menuid*. *item* kann folgende Werte annehmen:

SMIS_LEFTEDGE	-> LeftEdge
SMIS_TOPEDGE	-> TopEdge
SMIS_WIDTH	-> Width
SMIS_HEIGHT	-> Height
SMIS_FLAGS	-> Flags
SMIS_MUTUALEXCLUDE	-> MutualExclude
SMIS_COMMAND	-> Command
SMIS_NEXTSELECT	-> NextSelect

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_GetMenuItemStruct

wert = R_GetMenuItemStruct(appladdr, menu, menuid, itemid, item)
D0 A0 A1 D0 D1 D2

```
ULONG       wert;     /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu *menu;    /* Die Adresse der Menü-Struktur */
oder
ULONG       menu;     /* Menu-ID */
WORD        menuid;   /* Objektnummer des Menüs */
WORD        itemid;   /* Objektnummer des Menüeintrages */
WORD        item;     /* zu ermittelnde Strukturkomponente */
```

Ermittelt den Wert der in *item* angegebenen Strukturkomponente der Menuitem-Struktur des Menüeintrages mit der Objektnummer *itemid* im Pull-Down-Menü mit der Objektnummer *menuid*.

item kann folgende Werte annehmen:

GMIS_STRUCT	-> Adresse der MenuItem-Struktur
GMIS_NEXTITEM	-> struct MenuItem *NextItem
GMIS_LEFTEDGE	-> LeftEdge
GMIS_TOPEDGE	-> TopEdge
GMIS_WIDTH	-> Width
GMIS_HEIGHT	-> Height
GMIS_FLAGS	-> Flags
GMIS_MUTUALEXCLUDE	-> MutualExclude
GMIS_ITEMFILL	-> APTR ItemFill
GMIS_SELECTFILL	-> APTR SelectFill
GMIS_COMMAND	-> Command
GMIS_SUBITEM	-> struct MenuItem *SubItem
GMIS_NEXTSELECT	-> NextSelect

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

R_GetMenuSubStruct

wert = R_GetMenuSubStruct(appladdr, menu, menuid, itemid, subid, item)
D0 A0 A1 D0 D1 D2 D3

```
ULONG       wert;    /* Rückgabewert */
APTR        appladdr; /* Rückgabewert der Funktion R_InitAppl */
struct Menu *menu;   /* Die Adresse der Menü-Struktur */
oder
ULONG       menu;    /* Menu-ID */
WORD        menuid;  /* Objektnummer des Menüs */
WORD        itemid;  /* Objektnummer des Menüeintrages */
WORD        subid;   /* Objektnummer des Submenüs */
WORD        item;    /* Strukturkomponente, die ermittelt werden soll */
```

Ermittelt den Wert der in *item* angegebenen Strukturkomponenten des Submenü mit der Objektnummer *subid*, dessen Ursprung im Menüeintrag mit der Objektnummer *itemid* liegt und sich im Pull-Down-Menü mit der Objektnummer *menuid* befindet.
item kann folgende Werte annehemen:

GMSS_STRUCT	-> Adresse der MenuItem-Struktur
GMSS_NEXTITEM	-> struct MenuItem *NextItem
GMSS_LEFTEDGE	-> LeftEdge
GMSS_TOPEDGE	-> TopEdge
GMSS_WIDTH	-> Width
GMSS_HEIGHT	-> Height
GMSS_FLAGS	-> Flags
GMSS_MUTUALEXCLUDE	-> MutualExclude
GMSS_ITEMFILL	-> APTR ItemFill
GMSS_SELECTFILL	-> APTR SelectFill
GMSS_COMMAND	-> Command
GMSS_NEXTSELECT	-> NextSelect

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeAPP definiert wurde.

Kompatibilitätsfunktionen

Die auf den nachfolgenden Seiten beschriebenen Funktionen sind aus Kompatibilitätsgründen noch in der Bibliothek enthalten und nur Vollständigheitshalber hier aufgeführt. Im Zusammenarbeit mit den neuen Funktionen sollten diese nicht mehr genutzt werden.

R_OldShowScreen

screen = R_OldShowScreen(appladdr, mod, depth, title);
D0 A0 D1 D2 A1

```
struct Screen   *screen;   /* Rückgabewert */
APTR            appladdr;  /* Rückgabewert der Funktion initappl */
LONG            mod;       /* Die Auflösung des neu zu öffnenden Screens /*
LONG            depth;     /* Die Bitplaneanzahl des Screens */
BYTE            *title;    /* Stringadresse des Screentitels */
```

Öffnet einen Screen in der angegebenen Auflösung und Tiefe und liefert die Adresse der Screen-Struktur zurück. Ist für den Workbech-Screen eine Overscan-Auflösung eingestell, wird diese berücksichtigt und der neue Screen ebenfalls im Overscan geöffnet.

Hinweis:
Unter Kickstart/Workbench 2.0 wird der Screen automatisch im 3-D Look geöffnet!

R_OldFormAlert

id = R_OldFormAlert(default, commandstring);
D0 D0 A0

```
WORD      id;          /* Rückgabewert */
LONG      default;     /* Gadget das direkt mit Return bestädigt werden kann */
BYTE      *commandstring; /* speziell formatierter Ausgabetext */
```

Zeigt einen Alarm-Requester zentriert auf dem aktiven Screen an. Im commandstring sind sowohl die Texte der Alarmmeldung als auch die der Gadgets enthalten. Die Anzahl und Breite der Textzeilen richtet sich nach den Trennsymbolen im Text. Ebenso die Anzahl der dargestellten Gadgets. Natürlich kann der Text nicht breiter als der Screen werden. In diesem Fall wird automatisch getrennt. Auch bei den Gadgets wird eine weitere Zeile eingefügt, wenn der im Gadget anzuzeigende Text zu lang wird.

Der String hat folgendes Format:

"[SYMBOLNUMMER][1. Textzeile|2. Textzeile|3. Text...|...][1. Gadget|2. Gadg...|...]"

Wie Sie sehen werden die Textzeile und die Gadgets durch ein |-Zeichen voneinander getrennt. Die erste Ziffer, die mit SYMBOLNUMMER bezeichnet ist, wählt das Symbol, das links im Requester erscheinen soll.

Bei der Symbolwahl haben Sie vier Möglichkeiten:

0-> kein Symbol
1-> NOTE; ein auf der Spitze stehendes Quadrat mit Ausrufezeichen
2-> WAIT; ei Fragezeichen innerhalb eines Achtung-Schildes
3-> STOP; ein Stop-Schild

Eines der Gadgets kann durch die Variable *default* als Default Gadget definiert werden. Das Default Gadget ist durch einen dicken Rahmen in der Farbe mit dem Index 3 (unter 2.0 normalerweise dunkelblau) unübersehbar markiert und kann mit der Taste ausgewählt werden. Ist der Requester auf dem Bildschirm angezeigt, kann der Rahmen mit der Cursor-Links- oder mit der Tab-Taste auf das rechts daneben und mit der Cursor-Rechts- oder Shift/Tab-Taste auf das links daneben befindliche Gadget gewchselt werden. Es ist imer das Gadget mit Return wählbar, auf dem der Default-Rahmen gezeichnet ist.
Ist die variable *default* gleich 0, wird kein Rahmen gezeichnet und es kann auch nicht, wie beschrieben, gewechselt werden.

R_OldFiler

error = R_OldFiler(filereqstruct);
D0 A0

WORD error; /* Rückgabewert */
struct RCTFileRequester *filereqstruct; /* Adresse der FileRequester-Struktur */

Mit Aufruf dieser Funktion erscheint eine komfortables, vollkommen tastatursteuerbares, reentrantes Dateiauswahlfenster.
Die Datenstruktur, deren Adresse übergeben werden muß, muß mit Null initialisiert werden, damit keine unbenutzten Felder mit Werten vorbelegt sind.

Die Datenstruktur und die einstellbaren Flags wurden vom FileRequester der Requester-Library übernommen, so daß eine Umstellung auf den RCT-FileRequester sehr einfach gemacht werden kann.

Die Struktur (Requester-Library) ist folgendermaßen aufgebaut:

```
struct RCTFileRequester
   {
   UWORD VersionNumber; /* Die Versionsnummer der Library, die beim Program-
   mieren verwendet wird */
   char *Title;  /* Der Text, der in der Titelleiste des Fensters angezeigt
   werden soll */
   char *Dir; /* Der Pfad, der angezeigt werden soll. Dies ist auch der
   Bufferspeicher für den kompletten Pfad, der zurückgeliefert wird. Er
   sollte ausreichend dimensioniert sein. */
   char *File;   /* Der Dateiname, der angezeigt werden soll. In diesen
   Buffer wird auch der ausgewählte Dateiname kopiert. Er sollte ebenfalls
   ausreichend dimensioniert sein. */
   char *PathName;   /* Ein Buffer in den Pfad und Dateiname kopiert werden
   können. */
   struct Window *Window;   /* Ein Fenster in dem der FileRequester angezeigt
   werden soll. Im allgemeinen wird der FileRequester auf dem Screen geöff-
   net, auf dem auch die System-Alerts erscheinen würden. Siehe Funktion
   setsysrequest für weitere Erläuterungen. */
   UWORD reserved;   /* wird ignoriert */
   UWORD numlines;   /* wird ignoriert */
   UWORD numcolumns; /* wird ignoriert */
   UWORD devcolumns; /* wird ignoriert */
   ULONG Flags;  /* siehe nachfolgende Liste */
   UWORD dirnamescolor; /* wird ignoriert */
   UWORD filenamescolor;   /* wird ignoriert */
   UWORD devicenamescolor; /* wird ignoriert */
   UWORD fontnamescolor;   /* wird ignoriert */
   UWORD fontsizecolor; /* wird ignoriert */
   UWORD detailcolor;   /* wird ignoriert */
   UWORD blockcolor; /* wird ignoriert */
   UWORD gadgettextcolor;   /* wird ignoriert */
   UWORD textmessagecolor; /* wird ignoriert */
   UWORD stringnamecolor;   /* wird ignoriert */
```

```
    UWORD stringgadgetcolor; /* wird ignoriert */
    UWORD boxbordercolor;    /* wird ignoriert */
     UWORD gadgetboxcolor;   /* wird ignoriert */
     UWORD FRU_Stuff[18]; /* wird ignoriert */
     struct DateStamp DirDateStamp; /* wird ignoriert */
     UWORD WindowLeftEdge;   /* Die beiden nächsten Felder werden benutzt, um*/
                             /* den Requester zu positionieren */
     UWORD WindowTopEdge;
     UWORD FontYSize; /* wird ignoriert */
     UWORD FontStyle; /* wird ignoriert */
     ULONG reserved1; /* wird ignoriert */
     char Hide[];  /* Stringgadget Eintrag für Dateinamen, die nicht in der*/
                   /* Liste angezeigt werden sollen. */
     char Show[];  /* Stringgadget-Eintra fpr Dateinamen, die unbedingt*/
                   /* angezeigt werden sollen. */
     WORD FileBufferPos; /* Anzeigewerte für Stringgadgets */
     WORD FileDispPos;
     WORD DirBufferPos;
     WORD DirDispPos;
     WORD HideBufferPos;
     WORD HideDispPos;
     WORD HideBufferPos;
     WORD ShowBufferPos;
     WORD HideDispPos;
/** die nachfilgenden Stukturteile sind veränderlich. Es darf niemals lesend
darauf zugegriffen oder deren Inhalte verändert werden */
     ...
     };
```

DSIZE ist die Definition für die Größe des Pfadnamen-Arrays und ist normalerweise mit mindestens 128 Byte groß.
FCHARS steht für die Größe des Dateinamen-Array und sollte eine Größe von 32 Byte nicht unterschreiten.
WILDLENGTH ist die Länge des Strings für den HIDE- und SHOW-Text und sollte ebenfalls eine Länge von 32 Byte nicht unterschreiten.

Mit den folgenden Flags kann das Aussehen und die Art des Requesters verändert werden:

FRQCACHINGM /* Die eingelesenen Dateinamen sollen direkt nach einem erneuten Aufruf wieder angezeigt werden */
FRQABSOLUTXYM /* Die Koordinaten in den Feldern WindowLeftEdge und WindowTopEdge werden berücksichtigt. Wird dieses Flag nicht gesetzt, werden die Koordinaten ignoriert */
FRQNOSORTM /* Die Dateien sollen unsortiert angezeigt werden. */
FRQSAVINGM /* Im OK-Gadget wird der Text für Speichern (je nach Nationalität z.b. Speichern) angezeigt. */
FRQLOADINGM /* IM OK-Gadget wird der Text für Laden (je nach Nationalität z.B. Laden) angezeigt. */
FRQNODEPTHM /* Das den FileRequester umschließende Fenster soll keine Gadgets besitzen, mit denen es in den Vorder oder Hintergrund geklickt werden kann. Bei Backdrop-Windows eine bevorzugte Einstellung */

R_WaitMsg

void R_WaitMsg(port, message);
A0 A1

struct MessagePort *port; /* Die Adresse des Messageports, von dem die Nachrichten empfangen werden sollen */
struct IntuiMessage message; /* Eine komplette Struktur, in der eine Kopie der angekommenen Nachricht abgelegt wird. */

Anders als bei der Funktion *R_ReqWaitMsg* werden bei dieser die IDCMP-Flags des Fensters nicht beeinflußt. Sie liefert eine exakte Kopie der Nachricht und schickt die Original-Nachricht an das System zurück. Die Auswertung bleibt Ihnen überlassen

Beispiel

```
struct IntuiMessage message; /* keinen Adresspointer sondern die gesamte
Struktur */
struct Window *window;
...
R_WaitMsg( window->UserPort, &message);
if( message.Class & RAWKEY)
    {
    ...
    }
```

R_ResAddr

adr = R_ResAddr(appladdr, art, id)
D0 A0 D0 D1

```
VOID  *adr;     /* Rückgabewert je nach Objektart */
APTR  appladdr; /* Rückgabewert der Funktion initappl */
WORD  art;      /* Objektart */
WORD  id;       /* Objektnummer */
```

Ermittelt die Adresse der in der Resource-Datei enthaltenen Objekte (Requester, Menüleisten...). Ab Version 1.6 der Library muß die Funkiton nicht mehr zwingend aufgerufen werden. Alle Funktione, die eine Requester- oder Menüadresse erwarten können auch mit der Requester- oder Menü-ID versorgt werden. Die Adresse wird dann automatisch ermittelt.

Folgende Objektarten werden z.Z. unterstützt:

Requester: 0
Menüleisten: 1

Die *id* der Objekte wird beim Speichern der MAKEAPP-Dateien in einer Zusatzdatei abgelegt, die nur Konstanten-Definitionen enthält. Die Vorgabe kann in MakeApp unter dem Menüpunkt Ausgabe eingestellt werden.

Hinweis:
Hierbei handelt es sich um eine Funktion die ebenfalls aus Kompatibilitätsgründen noch enthalten ist.

Beispiel:

```
#define TESTOB 3 /* Requester */

void funk( void)
{
struct Reqester *req;
req = R_ResAddr( ApplAddr, 0, TESTOB);
...
}
```

R_OldRctLoad

rctaddr = R_OldRctLoad(pfad)
D0 A0

```
APTR        rctaddr;   /* Adresse der geladenen Datei */
UBYTE       *pfad;     /* die Adresse eines Strings */
```

Lädt die in Pfad angegebene Datei in den Speicher und liefert deren Adresse zurück. In *pfad* sollte nur der Dateiname angegeben werden, da sich andernfalls die Datei zwingend an dem durch den Pfad bestimmten Ort befinden muß. Geben Sie nur den Dateinamen an, wird zuerst im aktuellen Verzeichnis und dann im Verzeichnis mit dem Namen RCT: gesucht. Schlägt die Suche fehl, wird die Datei-Auswahl angezeigt und Sie können die RCT-Datei selbst lokalisieren.

Änderungen/Neuerungen:
In den Requester-Flags wird das Flag NOISYREQ generell gesetzt. Dadurch besteht die Möglichkeit, Rawkey-Codes auch bei angezeigtem Requester zu empfangen. Benutzen Sie zum Empfangen der Nachrichten die Funktion *R_WaitMsg*. Die Funktion *R_ReqWaitMsg* kann nur GADGETDOWN und GADGETUP Nachrichten liefern.

Hinweis:
Hierbei handelt es ich um eine alte Funktion, die lediglich aus Kompatibilitätsgründen noch enthalten ist. Benutzen Sie besser die Funktion *R_ResLoad*.

R_OldMenuDraw

error = R_OldMenuDraw(appladdr, menu, window)
D0 A0 A1 A2

```
WORD            error;     /* Rückgabewert */
APTR            appladdr;  /* Rückgabewert der Funktion R_InitAppl */
struct Menu     *menu;     /* Die Adresse der Menü-Struktur */
oder
ULONG           menu;      /* Menu-ID */
struct Window *window;     /* Die Adresse des Fensters */
```

Setzt in das in *window* angegebene Window die in *menu* angegebene Menüleiste.

Änderungen:
Der neuen Funktion muß nicht unbedingt die Adresse des Menüs übergeben werden sondern es genügt die Übergabe der Menü-ID, wie sie in MakeApp definiert wurde.

Anhang

Fehlermelungen der Library:

R_ResLoad

0	RLOAD_NOPARMS	Fehlerhafte Parameterliste
1	RLOAD_NOMEM1	Keinen Speicher für Load-Kontrollstruktur
2	RLOAD_NOFILE	Keine Datei zum Laden gefunden
3	RLOAD_READERR1	Fehler beim laden des Kontrollstrings
4	RLOAD_NORES	Keine RCT-Datei
5	RLOAD_READERR2	Fehler beim Laden des Dateiheaders
6	RLOAD_NOMEM2	Keinen Speicher um Datei zu laden
7	RLOAD_NOCHIP1	Keinen Speicher für Imagedaten
8	RLOAD_READERR3	Fehler beim Laden der RCT-Datei
9	RLOAD_NOSTRMEM	Fehler beim Allocieren des Speichers für die Intuitexte
10	RLOAD_EXTFNTERR	Fehler beim Laden oder Allocieren von Disk-Fonts
11	R_NOTAGADR	Keine TagItemliste übergeben
12	R_NOAPPLBASE	Inkompatibler Aufruf. Rückgabeadresse der Funktion R_InitAppl fehlerhaft
13	RXLOAD_NOMEM1	Keinen Speicher für Listenstruktur
14	RXLOAD_FAILED	Fehler beim Laden

R_ResFree

15	RFREE_ERR	RCT-Adresse zum Freigeben nicht in der Liste gefunden
12	R_NOAPPLBASE	Inkompatibler Aufruf. Rückgabeadresse der Funktion R_InitAppl fehlerhaft

R_ResAddr

12	R_NOAPPLBASE	Inkompatibler Aufruf. Rückgabeadresse der Funktion R_InitAppl fehlerhaft
16	RADDR_IDOVER	ID des gewünschten Objektes ist größer als die, den höchsten in der Datei enthaltenen.

R_ChangeAccess

12	R_NOAPPLBASE	Inkompatibler Aufruf. Rückgabeadresse der Funktion R_InitAppl fehlerhaft.
17	RCHANGE_NOADR	Gewünschte Datei nicht gefunden.

R_ReqDraw

18	**RDRAW_NOREQOP**	Fehler beim Anzeigen des Requesters
19	**RDRAW_NOLISTMEM**	Fehler beim Allocieren des Listenspeichers.

R_ReqEnd

20	**REND_NOLISTOBJ**	Objekt nicht in der Objektliste gefunden.

R_ReqDrawWindow

21	**RDRAWW_REQTOBIG**	Requester zu Groß um ihn auf dem Screen anzuzeigen.
22	**RDRAWW_NOWINMEM**	Kann nur auftreten, wenn das Programm nicht an die Library V16 angepasst ist.
23	**RDRAWW_NOWINOP**	Fenster konnte nicht geöffnet werden.
24	**RDRAWW_NOREQOP**	Requester konnte nicht geöffnet werden.

R_ShowScreen

25	**ROPSC_NOMEM**	Kein Speicher für NewScreen-Struktur.
26	**ROPSC_NOSCR**	Screen konnte nicht geöffnet werden.
27	**ROPXSC_NOMEM**	Keinen Speicher für Screenliste.
28	**POPXSC_NOSCR**	Fehler beim Öffnen des Screens.

R_RemScreen

29	**RREMSC_UNKOWN**	Angegebener Screen unbekannt.
30	**RREMSC_WINUNKNOWN**	Fremdes Fenster auf dem Screen.
31	**RREMSC_NOSCR**	Angegebener Screen nicht in der Liste enthalten.

R_Showindow

32	**RSWIN_NOMEM**	Keinen Speicher für Windowliste.
33	**RSWIN_NOWINOP**	Fehler beim Öffnen des Fensters.

R_RemWindow

34	**RREMW_WINUNKNOWN**	Angegebenenes Fenster unbekannt.

Anpassung von alten Programmen

Die wichtigste Änderung ist die Einführung der beiden neuen Funktionen **R_InitAppl** und **R_ExitAppl**.

Eine weitere wichtige Änderung betrifft die Übergabe der Requester- und Menüadresse an die Funktionen. Ab der Version 1.6 der Library ist es möglich, den Funktionen, denen bislang die Requester- oder Menüadressen übergeben werden mußte, einfach die Requester- oder Menü-ID zu übergeben.
Die alte Funktionsweise der Library ist allerdings weiterhin erhalten geblieben.

Anpassung ihrer Programm an die neue Bibliothek.

Wenn Sie bereits die Funktionen der alten Bibliothekgenutzt haben und jetzt die neu hinzugekommenen Fähigkeiten nutzten wollen, sollten Sie ihre Programme unbedingt anpassen. Auch um auf zukünftige Neuerungen vorbereitet zu sein, ist eine Anpassung notwendig.

Nach dem öffnen der Library sollte als erstes die Funktion **R_InitAppl** aufgerufen werden. Diese reserviert Speicher für einige Parameter, bereitet eine Default-NewWindow- und -NewScreen-Struktur vor. Die von der Funktion **R_InitAppl** zurückgelieferte Adresse muß unbedingt jeder Funktion, die bislang, die RückgabeAdresse der Funktion rctload (in der alten Dokumentation mit **rctadr** beschrieben) erwartet hat, übergeben werden. Sie sollten also die Funktion **rctload** nicht mehr verwenden. Vielmehr haben Sie jetzt die Möglichkeit mit der Funktion **R_ResLoad**, mehrere RCT-Dateien zu laden und erhalten statt einer Adresse auf die Datei nur noch ein unkompliziertes Datei-Handle zurückgeliefert. Die Adresse der zuletzt geladenen Datei wird automatisch in der durch die Funktion **R_InitAppl** reservierten Parameter- und Variablen-Speicher gemerkt. Jeder Funktionszugriff, der etwas mit Requestern, Gadgets oder Menüleisten zu tun hat, bezieht sich auf die zuletzt geladene Datei. Will man mehrere Dateien laden, können die Zugriffsrechte mittels der Funktion **R_ChangeAccess** und Übergabe des von der Funktion **R_ResLoad** zurückgerhaltenen Datei-Handles geändert werden. Will man sich auch den Datei-Handle nicht merken, besteht die Möglichkeit, das ganze auch durch Übergabe des Dateinamens an die Funktion **R_ChangeAccess** zu ändern.
Die Funktion **rctaddr**, zum Ermitteln der Adressen von Requester- und Menü-Strukturen muß nicht unbedingt aufgerufen werden. Allen Funktionen, denen in der alten Bibliothek noch eine Strukturadresse übergeben werden mußte, können jetzt auch die Requester- oder Menü-ID übergeben werden, die Sie sonst der Funktion **rctaddr** mitteilen mußten.

Einige Funkionen wurden stark erweitert und benötigen mehr Parameter als zuvor. Die alten Aufrufe sind lediglich aus Kompatibilitätsgründen erhalten geblieben. Es sind dies die Funktionen:

R_ResLoad	<-	rctload
R_MenuDraw	<-	menudraw
R_ShowScreen	<-	showscreen
R_FormAlert	<-	form_alert
R_Filer	<-	filer

Durch die Anpassung Ihrer Programm an die neuen Funktionen gewährleisten Sie, daß Sie jederzeit das Programm beenden können, ohne sich über noch geöffnete Requester, Menüs, Fenster und Screens gedanken machen müssen. Das Schließen und Freigeben der angeforderten Ressourcen (Hilfsmittel) übernimmt die Funktion **R_ExitAppl**. Sie müssen dieser lediglich die von der Funktion **R_InitAppl** gelieferte Adresse übergeben.

Index

Symbole

3-D Look 10

C

CUSTOMBITMAP 15
CUSTOMSCREEN 15

D

Dynamische Einträge 34

F

Farbanpassung 10

H

HIRES 16
HIRESILACE 16

I

IDCMP_ACTIVEWINDOW 24
IDCMP_CLOSEWINDOW 23
IDCMP_DELTAMOVE 24
IDCMP_DISKINSERTED 23
IDCMP_DISKREMOVED 23
IDCMP_GADGETDOWN 23
IDCMP_GADGETUP 23
IDCMP_INACTIVEWINDOW 24
IDCMP_INTUITICKS 24
IDCMP_MENUPICK 23
IDCMP_MENUVERIFY 15, 23
IDCMP_MOUSEBUTTON 25
IDCMP_MOUSEBUTTONS 23
IDCMP_MOUSEMOVE 23
IDCMP_NEWPREFS 23
IDCMP_NEWSIZE 23
IDCMP_RAWKEY 23
IDCMP_REFRESHWINDOW 23
IDCMP_REQCLEAR 23
IDCMP_REQSET 23
IDCMP_REQVERIFY 23

IDCMP_SIZEVERIFY 23
IDCMP_VANILLAKEY 24

O

OSCAN_STANDARD 18
OSCAN_TEXT 18
OSCAN_VIDEO 18
OSERR_NOCHIPS 16
OSERR_NOMEM 16

R

R_ActiveGad 68
R_Border 31
R_BUMERANG_STRINGS 34
R_ChangeAccess 54
  Rres_Handle 54
  Rres_Name 54
R_ChangeAccessTags 54
R_Create 31
  R_Border 31
  R_Dup_Res 31
  R_Gadget 31
  R_Menu 31
  R_MenuItem 31
  R_Request 31
  R_Text 31
  RNo_Brd 32
  RNo_BrdData 32
  RNo_Img 32
  RNo_ImgData 32
  RNo_IntTxt 31
  RNo_IntTxtStr 32
  RNo_TxtAttr 31
R_CreateTags 31
R_Destroy 32
R_Dup_Res 31
R_ExitAppl 12
R_Filer 33
  R_BUMERANG_STRINGS 34
  RF_AcceptPattern 38
  RF_DoMultiSelect 37
  RF_DoPatterns 37
  RF_DoSaveMode 37
  RF_DrawersOnly 37
  RF_FilterDrawers 38
  RF_FilterFunc 37
  RF_Flags1 37
  RF_Flags2 37

RF_Handle 34
RF_HookFunc 38
RF_InitialDrawer 36
RF_InitialFile 36
RF_InitialHeight 36
RF_InitialPattern 36
RF_InitialTopEdge 36
RF_InitialWidth 36
RF_InitioalLeftEdge 36
RF_IntuiFunc 35
RF_Locale 36
RF_NegativeText 36
RF_OverWriteCheckMsg 38
RF_PositivText 36
RF_PrivateIDCMP 35
RF_PubScreenName 35
RF_RejectIcons 37
RF_RejectPattern 37
RF_Screen 35
RF_SleepWindow 35
RF_TextAttr 35
RF_TitleText 36
RF_UserData 35
RF_Window 34
R_FilerCacheFree 38
RF_Handle 39
R_FormAlert 40
RFA_AlertText 41
RFA_BeginSoundSample 41
RFA_BeginSpeek 41
RFA_Center 43
RFA_CloseID 42
RFA_DefaultID 41
RFA_DefaultTimeout 41
RFA_EndSoundSample 41
RFA_EndSpeek 42
RFA_GadgetText 41
RFA_Human 42
RFA_Image 42
RFA_ImageData 42
RFA_ImageToggle 42
RFA_OnlySoundSample 41
RFA_OnlySpeek 42
RFA_Timeout 41
RFA_TimeoutSoundSample 41
RFA_Window 43
RFA_WindowClose 42
RFA_WindowDepth 42
RFA_WindowDrag 42
RFA_WindowFlags 42
R_GadDrawWindow 10, 58

R_Gadget 31
R_GetGadStruct 65
R_GetGadText 64
R_GetMenuItemStruct 76
R_GetMenuIText 74
R_GetMenuSubStruct 77
R_GetReqStruct 61
R_GetReqText 63
R_GetStrGadStruct 70
R_GetStrGadText 69
R_InitAppl 9, 21
RIA_AutoRenderColor 10
RIA_Debug 11
RIA_GlobIPort 11
RIA_NoShortCuts 11
RIA_SameIDCMP 10
RIA_Version 10
R_InitApplTags 9
R_Menu 31
R_MenuDraw 72
RMD_Justify 72
R_MenuDrawTags 72
R_MenuEnd 73
R_MenuItem 31
R_Raw2Asc 29
R_RedrawGad 67
R_RemScreen 19
R_RemWindow 28
R_ReqCenter 56
R_ReqDraw 10, 56
R_ReqDrawWindow 10, 57
R_ReqEnd 59
R_Request 31
R_ReqWaitMsg 60
R_ResAddr 83
R_ResetSysRequest 30
R_ResFree 53
R_ResLoad 51
Rres_Address 51
Rres_LocalFlags 52
Rres_LocalPath 52
Rres_Name 51
R_ScreenModeReq 44
R_SetGadStruct 66
R_SetGadText 64
R_SetMenuItemStruct 75

R_SetMenuTitle 73
R_SetPropGadStruct 71
R_SetReqStruct 62
R_SetReqText 63
R_SetStrGadText 68
R_SetSysRequest 29
R_ShowScreen 13
CUSTOMBITMAP 15
CUSTOMSCREEN 15
RSA_AutoScroll 18
RSA_Behind 18
RSA_BitMap 16
RSA_BlockPen 15
RSA_Colors 17
RSA_DClip 18
RSA_Depth 14
RSA_DetailPen 15
RSA_DisplayID 16
RSA_DisplayIDStruct 16
RSA_ErrorCode 16
OSERR_NOCHIPS 16
OSERR_NOMEM 16
OSERR_NOMONITOR 16
RSA_Font 15
RSA_FullPalette 19
RSA_Height 14
RSA_Left 14
RSA_LikeWorkbench 14
RSA_Overscan 18
OSCAN_STANDARD 18
OSCAN_TEXT 18
OSCAN_VIDEO 18
RSA_Pens 19
RSA_PubName 17
RSA_PubSig 17
RSA_PubTask 17
RSA_ShowTitle 18
RSA_SysFont 17
RSA_Title 16
RSA_Top 14
RSA_Type 15
RSA_Width 14
SCREENBEHIND 15
SCREENQUIT 15
R_ShowScreenTags 13
R_ShowWindow 21
RWA_Activate 26
RWA_AutoAdjust 26
RWA_BackDrop 26
RWA_BackFill 27

RWA_BlockPen 22
RWA_Borderless 26
RWA_Checkmark 25
RWA_CloseGadget 26
RWA_CustomScreen 25
RWA_DepthGadget 26
RWA_DetailPen 22
RWA_DragBar 26
RWA_Flags 24
WFLG_ACTIVATE 25
WFLG_BACKDROP 24
WFLG_BORDERLESS 25
WFLG_CLOSEGADGET 24
WFLG_DEPTHGADGET 24
WFLG_DRAGBAR 24
WFLG_GIMMEZEROZERO 25
WFLG_NOCAREREFRESH 25
WFLG_REPORTMOUSE 25
WFLG_RMBTRAP 25
WFLG_SIMPLE_REFRESH 24
WFLG_SIZEBBOTTOM 24
WFLG_SIZEBRIGHT 24
WFLG_SIZEGADGET 24
WFLG_SMART_REFRESH 24
WFLG_SUPER_BITMAP 24
RWA_Gadgets 25
RWA_GimmeZeroZero 27
RWA_Height 22
RWA_IDCMP 22
IDCMP_ACTIVEWINDOW 24
IDCMP_CLOSEWINDOW 23
IDCMP_DELTAMOVE 24
IDCMP_DISKINSERTED 23
IDCMP_DISKREMOVED 23
IDCMP_GADGETDOWN 23
IDCMP_GADGETUP 23
IDCMP_INACTIVEWINDOW 24
IDCMP_INTUITICKS 24
IDCMP_MENUPICK 23
IDCMP_MENUVERIFY 23
IDCMP_MOUSEBUTTONS 23
IDCMP_MOUSEMOVE 23
IDCMP_NEWPREFS 23
IDCMP_NEWSIZE 23
IDCMP_RAWKEY 23
IDCMP_REFRESHWINDOW 23
IDCMP_REQCLEAR 23
IDCMP_REQSET 23
IDCMP_REQVERIFY 23
IDCMP_SIZEVERIFY 23
IDCMP_VANILLAKEY 24

RWA_InnerHeight 27
RWA_InnerWidth 27
RWA_Left 21
RWA_MaxHeight 26
RWA_MaxWidth 26
RWA_MinHeight 26
RWA_MinWidth 25
RWA_NoCareRefresh 26
RWA_PubScreen 27
RWA_PubScreenFallBack 27
RWA_PubScreenName 27
RWA_ReportMouse 26
RWA_RMBTrap 26
RWA_SCRCPY 21
RWA_ScreenTitle 27
RWA_SimpleRefresh 26
RWA_SizeBBottom 27
RWA_SizeBRight 26
RWA_SizeGadget 26
RWA_SmartRefresh 26
RWA_SuperBitMap 25
RWA_Title 25
RWA_Top 21
RWA_Width 22
RWA_Zoom 27
R_ShowWindowTags 21
R_Text 31
R_WaitMsg 82
RF_AcceptPattern 38
RF_DoMultiSelect 37
RF_DoPatterns 37
RF_DoSaveMode 37
RF_DrawersOnly 37
RF_FilterDrawers 38
RF_FilterFunc 37
RF_Flags1 37
RF_Flags2 37
RF_Handle 34, 39
RF_HookFunc 38
RF_InitialDrawer 36
RF_InitialFile 36
RF_InitialHeight 36
RF_InitialPattern 36
RF_InitialTopEdge 36
RF_InitialWidth 36
RF_InitioalLeftEdge 36
RF_IntuiFunc 35
RF_Locale 36

RF_NegativeText 36
RF_OverWriteCheckMsg 38
RF_PositivText 36
RF_PrivateIDCMP 35
RF_PubScreenName 35
RF_RejectIcons 37
RF_RejectPattern 37
RF_Screen 35
RF_SleepWindow 35
RF_TextAttr 35
RF_TitleText 36
RF_UserData 35
RF_Window 34
RFA_AlertText 41
RFA_BeginSoundSample 41
RFA_BeginSpeek 41
RFA_Center 43
RFA_CloseID 42
RFA_DefaultID 41
RFA_DefaultTimeout 41
RFA_EndSoundSample 41
RFA_EndSpeek 42
RFA_GadgetText 41
RFA_Human 42
RFA_Image 42
RFA_ImageData 42
RFA_ImageToggle 42
RFA_OnlySoundSample 41
RFA_OnlySpeek 42
RFA_Timeout 41
RFA_TimeoutSoundSample 41
RFA_WindowClose 42
RFA_WindowDepth 42
RFA_WindowDrag 42
RFA_WindowFlags 42
RIA_AutoRenderColor 10
RIA_Debug 11
RIA_GlobIPort 11
RIA_NoShortCuts 11
RIA_SameIDCMP 10
RIA_Version 10
RMD_Justify 72
RNo_Brd 32
RNo_BrdData 32
RNo_Img 32

RNo_ImgData 32
RNo_IntTxt 31
RNo_IntTxtStr 32
RNo_TxtAttr 31
RREMW_WINUNKNOWN 19
Rres_Address 51
Rres_Handle 54
Rres_LocalFlags 52
Rres_LocalPath 52
Rres_Name 51, 54
RSA_AutoScroll 18
RSA_Behind 18
RSA_BitMap 16
RSA_BlockPen 15
RSA_Colors 17
RSA_DClip 18
RSA_Depth 14
RSA_DetailPen 15
RSA_DisplayID 16
RSA_DisplayIDStruct 16
RSA_ErrorCode 16
RSA_Font 15
RSA_FullPalette 19
RSA_Height 14
RSA_Left 14
RSA_LikeWorkbench 14
RSA_Overscan 18
RSA_Pens 19
RSA_PubName 17
RSA_PubSig 17
RSA_PubTask 17
RSA_ShowTitle 18
RSA_SysFont 17
RSA_Title 16
RSA_Top 14
RSA_Type 15
RSA_Width 14
RWA_Activate 26
RWA_AutoAdjust 26
RWA_BackDrop 26
RWA_BackFill 27
RWA_BlockPen 22
RWA_Borderless 26
RWA_Checkmark 25
RWA_CloseGadget 26

RWA_CustomScreen 25
RWA_DepthGadget 26
RWA_DetailPen 22
RWA_DragBar 26
RWA_Flags 24
RWA_Gadgets 25
RWA_GimmeZeroZero 27
RWA_Height 22
RWA_IDCMP 22
RWA_InnerHeight 27
RWA_InnerWidth 27
RWA_Left 21
RWA_MaxHeight 26
RWA_MaxWidth 26
RWA_MinHeight 26
RWA_MinWidth 25
RWA_NoCareRefresh 26
RWA_PubScreen 27
RWA_PubScreenFallBack 27
RWA_PubScreenName 27
RWA_ReportMouse 26
RWA_RMBTrap 26
RWA_SCRCPY 21
RWA_ScreenTitle 27
RWA_SimpleRefresh 26
RWA_SizeBBottom 27
RWA_SizeBRight 26
RWA_SizeGadget 26
RWA_SmartRefresh 26
RWA_SuperBitMap 25
RWA_Title 25
RWA_Top 21
RWA_Width 22
RWA_Zoom 27

S

SCREENBEHIND 15, 18
SCREENQUIET 18
SCREENQUIT 15
Statische Einträge 34
struct ColorSpec 17
struct Image * 25
struct R_IntuiMessage 22

T

Tastaturkürzel 50

W

WFLG_ACTIVATE 25
WFLG_BACKDROP 24
WFLG_BORDERLESS 25
WFLG_CLOSEGADGET 24
WFLG_DEPTHGADGET 24
WFLG_DRAGBAR 24
WFLG_GIMMEZEROZERO 25
WFLG_NOCAREREFRESH 25
WFLG_REPORTMOUSE 25
WFLG_RMBTRAP 15, 25
WFLG_SIMPLE_REFRESH 24
WFLG_SIZEBBOTTOM 24
WFLG_SIZEBRIGHT 24
WFLG_SIZEGADGET 24
WFLG_SMART_REFRESH 24
WFLG_SUPER_BITMAP 24